APPLICATIONS OF AERODYNAMIC FORCES FOR SPACECRAFT ORBIT MANEUVERABILITY IN OPERATIONALLY RESPONSIVE SPACE AND SPACE RECONSTITUTION NEEDS

THESIS

Matthew N. Goodson, 1st Lt, USAF

AFIT/GA/ENY/12-M09

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GA/ENY/12-M09

APPLICATIONS OF AERODYNAMIC FORCES FOR SPACECRAFT ORBIT MANEUVERABILITY IN OPERATIONALLY RESPONSIVE SPACE AND SPACE RECONSTITUTION NEEDS

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Astronautical Engineering

Matthew N. Goodson, BS

1st Lieutenant, USAF

March 2012

AFIT/GA/ENY/12-M09

APPLICATIONS OF AERODYNAMIC FORCES FOR SPACECRAFT ORBIT MANEUVERABILITY IN OPERATIONALLY RESPONSIVE SPACE AND SPACE RECONSTITUTION NEEDS

Matthew N. Goodson, BS

1st Lieutenant, USAF

Approved:

Jonathan T. Black, PhD (Chairman)	Date
Kerry D. Hicks, PhD (Member)	Date
Ronald J. Simmons, PhD (Member)	Date

Abstract

Each year multiple satellites are launched to provide end users key pieces of information. This information ranges from remote sensing data for military or civilian purposes (weather forecasting, troop movements, agriculture production, etc.) to large bandwidth telecommunication sensors. No matter the type of information needed, society is demanding more. Because of this continual rise in information needs, the current model of launching one satellite for one mission is not sustainable. In order to satisfy the information needs of nations across the globe, a means for satellites to transition from one collection opportunity to another must be developed. One means of transitioning from collection opportunities involves using the aerodynamic forces experienced in the upper atmosphere to maneuver the spacecraft.

This research involves the use of aerodynamic forces on a spacecraft to conduct in-plane and out-of-plane maneuvers. It is assumed a satellite can use a small thruster to maintain an altitude within the upper atmosphere and use aerodynamic forces to conduct maneuvers. Comparisons will be made between satellites with nominal small force thrusters and satellites utilizing an aerodynamic design. Key focus areas will be the amount of fuel saved for similar maneuvering profiles and the amount of orbital changes possible. This study will use the Gaussian Variation of Parameter equations to calculate the thrust, aerodynamic, and orbital perturbations in a MATLAB code designed for modeling the space environment.

To my Wife

Acknowledgements

I would like to express my sincere appreciation to my research advisor, Dr. Jonathan Black, for his guidance and support throughout the course of this thesis effort. The insight and experience was certainly appreciated.

I would also like to thank my wife, parents, and brother for all the support and guidance they provided for not only this thesis but throughout my life. Without their guidance and love I would not be in life where I am at today.

Matthew N. Goodson

Table of Contents

Page

List of Figures

Page

List of Tables

APPLICATIONS OF AERODYNAMIC FORCES FOR SPACECRAFT ORBIT MANEUVERABILITY IN OPERATIONALLY RESPONSIVE SPACE AND SPACE RECONSTITUTION NEEDS

Introduction

General Issue

Space continues to be one of the greatest force multipliers for a nation or organization. Russia's launch of Sputnik in 1957 opened the world to an era of instant communication and remote sensing uses of which mankind has yet to reach the limit. Each year more and more satellites are launched to provide information to end-users. This information ranges from remote sensing data for military or civilian purposes (weather forecasting, troop movements, agriculture production, etc.) to large bandwidth telecommunication highways. No matter the type of information, the world's nations are demanding more.

Problem Statement

Due to the continual increase in information need combined with the demand for more rapid deployment of new information technology, the current model of launching one satellite for one mission is not sustainable. The current model is also leading to the congestion of space as more and more satellites are launched into orbit. In order to satisfy the information needs of the United States and other nations, a new standard in satellite mission capabilities is required. This standard must incorporate a means for satellites to transition from collect opportunities by efficiently changing orbits. This action will ultimately cut down the number of satellites required as well as allow for the quick turnaround of information. A secondary benefit of providing a means to change orbits is to allow for the economical cleanup of space. A satellite capable of moving from one orbit to another could efficiently collect space debris. The appropriate disposal of space debris would lead to the decreased likelihood of debris collisions with operating spacecraft. Operationally Responsive Space (ORS) and Space Reconstitution (SR) respectively are the two titles given to the missions mentioned above.

It is the opinion of the current satellite launch and operation models are unsustainable. By making satellites more maneuverable, the number of satellites needed in a constellation and re-tasking can be brought down. The reduction in number of launches and satellites needed for a mission is one of the main objectives of ORS missions. Many avenues of achieving the goals of ORS have been discussed. Most, however, focus on the cheap and easy to manufacture aspect using such ideas as common spacecraft buses for various missions and using plug-and-play components to fit the desired mission.

The second mission, SR, recently received a lot of attention on 10 February 2009 as the U.S. Iridium 33 communications satellite and defunct Russian military communications satellite Cosmos 2251 collided, creating two large debris clouds that caused a significant danger to operational satellites in low earth orbit for many years (Malik, 2009:1). As more and more spacecraft are put into orbit there is an increased probability of collisions; because more nations and organizations are creating their own space programs this rate is increasing exponentially. In order to provide future access to space new ways to de-orbit satellites quickly and safely and allow for the removal of debris left over from launches and defunct satellites must be established.

Proposed Solution

One such way of doing both ORS and SR missions is what has been named the space plane. A space plane uses the hypersonic speeds of the free stream air around it to produce lift. The X-37, X-40, and X-38 are all examples of space planes. Ultimately a space plane can be any satellite which uses some type of airfoil in the hypersonic flow to maneuver in Earth's atmosphere. These airfoils can be used for orbit maintenance by providing a simple lift vector normal to the orbit's velocity vector or a change in the velocity vector itself, leading to a change in orbit. The U.S. Department of Defense is interested in space plane capabilities and currently has requirements for their own space plane described in AFSPC Mission Needs Statement 001-97 "Tactical Military Operations in Space." For a complete tabular breakdown of the DoD's space plane requirements see Appendix A. A quick overview of the requirements is detailed below.

The general mission capabilities of the space plane taken from AFSPC Mission Needs Statement 001-97 are threefold:

> a. The Military Space plane System shall be capable of supporting a wide range of military air and space superiority, global attack, precision engagement, and information superiority missions requiring flight operations in, through, and from space and the trans-atmosphere.
>
> b. The Military Space plane System shall be capable of ascending to, operating in, and descending from designated orbits. The Military Space

plane System shall be capable of suborbital flight including exo-atmospheric flight.

c. The Military Space plane System shall be capable of carrying a payload and deploying or otherwise utilizing the payload to execute the military missions in orbit, while ascending to orbit, while descending from orbit, or during suborbital flight.

Using these general mission capabilities the space plane is expected to operate in four different design mission references;

1. Pop up into the exo-atmosphere from a sub-orbital flight and deliver a payload
2. Launch directly into an orbital trajectory with a payload
3. Launch into an orbital trajectory and land after one orbit at its takeoff base
4. Ferry the flight segment of the space plane from one location to another a minimum distance of 2000 nautical miles without landing

Because of this renewed emphasis in hypersonic air/space craft by the DoD, the area of hypersonics has seen an increased interest in the past decade and many advances have been made. Most research has focused on the basics of using space planes to change orbit inclination and orbit raising. This thesis will look further into the orbital maintenance regime and determine the practicality of using a space plane design to prolong the lifetime of the satellite orbit with and without electrical propulsion assistance.

Investigative Questions

A basic and quick example of using lift to change a spacecraft's orbit as well as an orbit maintainer can be shown using the basic equation for lift and drag displayed below. The two main variables a wing designer can change in producing a lift force are the lift coefficient and wing surface area. The density and velocity are variables of the space plane's orbit.

$$L = \frac{1}{2}\rho C_l A V^2 \tag{1}$$

$$D = \frac{1}{2}\rho C_d A V^2 \tag{2}$$

where C_l is the lift coefficient
C_d is the drag coefficient
A is the reference area
ρ is the density of the fluid the object is traveling in (commonly air)
V is the velocity of the object through the fluid

For common airfoils seen on today's sub-orbital airplanes, the lift coefficient typically has values ranging from 0 to 2 and does not produce a significant change to the lift force when compared with a change in wing surface area, a value ranging from 5 m^2, a hang glider, to 845 m^2, the Airbus A380. Based on this, changing the surface area of a wing is much more practical and will be used as the independent variable in the example. Assuming an airfoil travels through the atmosphere with a constant flight path angle of 10 degrees, the coefficients of drag and lift can be fixed. In this example the coefficient of drag is set to 2.25, a typical value for satellites and the coefficient of lift is set to 1, a number which will be explained later in this thesis (Carter, 2009:1). An initial altitude of 100 km, the corresponding air density using the 1976 Standard Atmosphere Model, a velocity of 7.5 km/s, and a constant mass of 1000 kg are used to fit with the orbits of satellites used in this thesis' discussion. Two important notes for the example are the assumptions that lift, drag, and weight being the only forces acting on the object and air continues to act as a fluid medium. Using Figure 1 below and summing the forces in the x and y directions, solving for the amount of surface area needed to keep an object with a 4 m^2 cross sectional area from losing height is possible. This value comes out to be 720 m^2. This surface area is only 100 m^2 less than that of the Airbus A380 and only has a mass approximately 0.33% of the unloaded take-off weight of the Airbus A380 (Airbus A380).

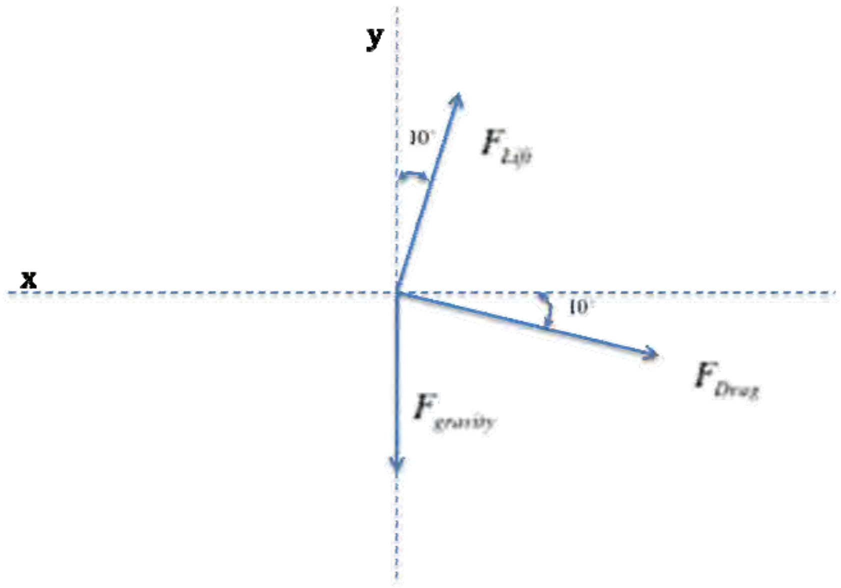


Figure 1: Space Plane Wing Analysis

Looking at this simple analysis one wonders why space planes should be considered at all. This thesis will show the problem is not as simple as summing the forces. Although building a wing large enough to sustain a satellite's orbit forever is not feasible; limited maintenance of the semi-major axis can be achieved.

A second key point in understanding the problem is showing how much change in velocity is needed in order to change a spacecraft's orbit inclination. This velocity change must come from some source, whether it is a conventional thruster or aerodynamic forces. Although inclination change is not discussed in this thesis, it is helpful in clarifying the high costs in velocity change needed to change a spacecraft's orbit. An inclination change in a spacecraft's orbit is commonly referred to as a simple plane change. The change in velocity required to go from one orbit inclination to another orbit inclination can be found by subtracting the first orbit's velocity from the second orbit's velocity as Figure 2 below shows.

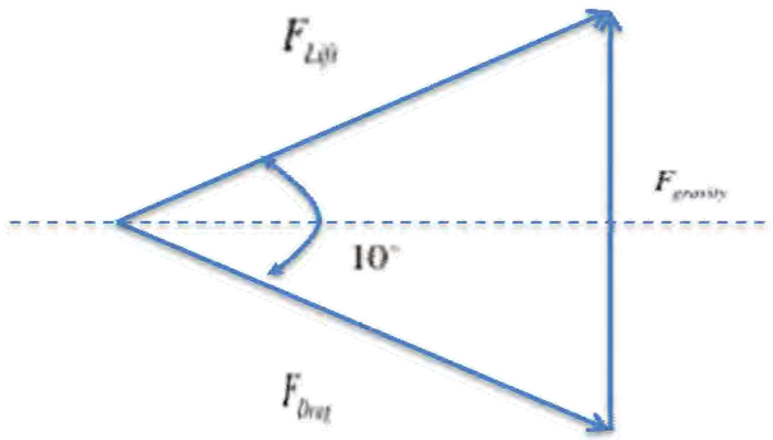


Figure 2: Simple Plane Change

By doing this subtraction the equation for a simple plane change becomes

$$\Delta V = 2V_i sin\left(\frac{\Delta i}{2}\right) \tag{3}$$

where V is the orbital velocity
i is the inclination

If an initial velocity of 7.5 km/s is assumed the change in velocity required for a given change in inclination can be graphed.

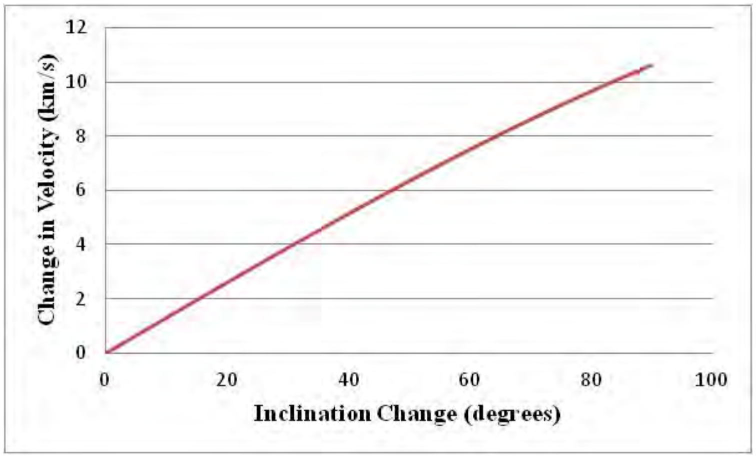


Figure 3: Inclination Change Cost

As Figure 3 shows the necessary change in velocity quickly adds up and after 60 degrees the velocity change needed equals the orbital velocity of the spacecraft. This large change in velocity is the main reason a spacecraft's orbit inclination is not changed repeatedly. To do so would require a significant amount of fuel. However, if we can use the Earth's atmosphere by itself or along with a constant thrust engine to produce a force a satellite can potentially change its orbit inclination while minimizing the altitude loss and thus the amount of onboard fuel used.

Methodology Overview

In order to fully address the problem described above and understand the usefulness of aerodynamic forces to help maintain satellite orbits, a foundation of understanding in four major areas is needed. These areas are orbital dynamics, Gaussian variation of parameters equations, orbital perturbations, and hypersonic aerodynamics. This understanding of useful areas will be accomplished in the methodology chapter of this paper.

Once these areas are discussed an overview of the steps taken in creating a model to study the effects on a satellite using a one Newton thruster without aerodynamic assistance and a satellite using the one Newton thruster coupled with aerodynamic forces will be discussed. The most important key points of this section will be how the equations derived from the four major areas discussed above are employed together and the necessary assumptions made in creating the spacecraft model.

The thesis then moves into the results and analysis chapter and discusses the capabilities of the various spacecraft mentioned above to maintain the perigee of their orbits. The scenario of perigee maintenance was chosen because it allows for immediate benefits for ORS and SR missions. Using the design considerations discussed above, an emphasis will be placed on the comparison of a spacecraft using only a thruster versus a spacecraft that uses aerodynamic forces with and without the thruster.

The final chapter will be used to discuss the effects a space plane and its capabilities will have on the spacecraft community. Recommendations of whether the spacecraft will be useful in the fields of operationally responsive space and space reconstitution will also be made. This thesis will show small satellites incorporating the designs discussed in this paper could significantly decrease the loss in orbit time compared with satellites using present day body designs.

Literature Review

Chapter Overview

The past decade has seen a large increase in hypersonic studies. Even though the study of hypersonics has been around since the 1950's the use of aero-assisted vehicles has only recently come into its own as a viable option for on-orbit maneuvering. Recent analysis has shown the concept of using aerodynamic forces to maneuver a vehicle can provide substantial savings in fuel when coupled with other thrusting methods (Jolley, 2001:1). The studies in hypersonics have also proven the simple calculations used earlier, Equation 1 and Equation 2, for finding the necessary lift to maintain an orbit are not as simple as shown. This chapter will discuss the research, which has been conducted in the area of hypersonics to provide a better understanding of the complications and differences hypersonic velocity creates.

Relevant Research

At the 100 km height chosen in the analysis above, many assumptions and simple equations break down when describing the effects of air on a vehicle. At altitudes above 85 km the density of the atmosphere becomes much thinner (rarification) and an alternate method using dynamic pressure is needed in order to calculate coefficients of lift and drag. One of the more common approaches is Newtonian theory. Using this approach for an infinitely thin flat plate and noting L/D is equal to the cotangent of the angle of attack produces the blue solid line shown in Figure 4. The red line includes the effects of laminar skin friction for a surface with a Reynolds number equal to $3x10^4$ and a Mach number of 10.

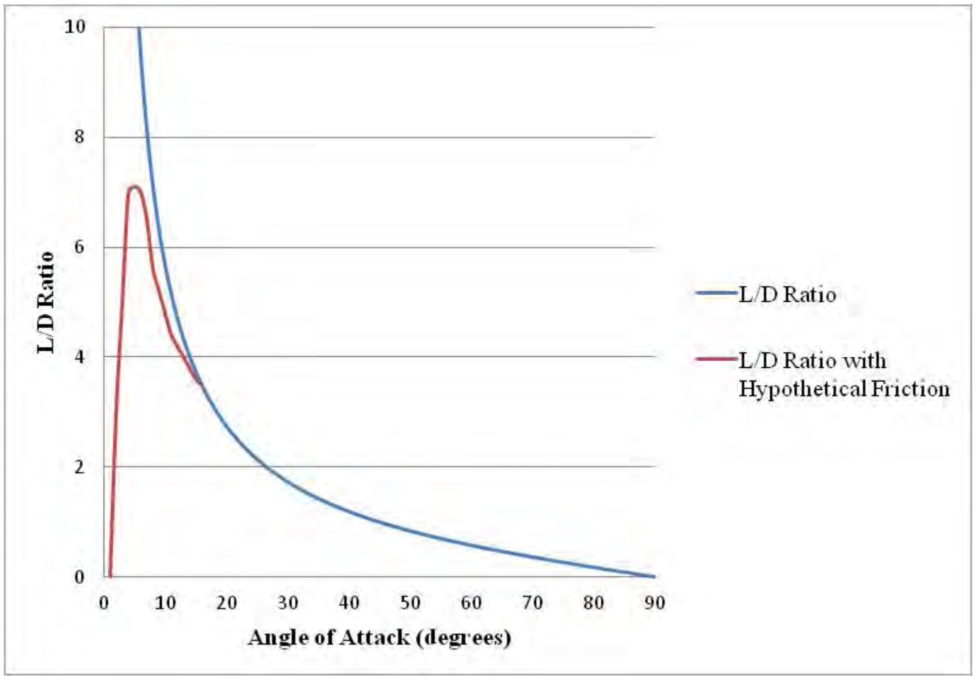


Figure 4: Newtonian Results for a Flat Plate

As the reader can see the L/D ratio does not approach infinity and has a maximum ratio associated with the laminar skin friction. Since the infinitely thin flat plate is the most efficient lifting surface, one can conclude the desired L/D ratios of hypersonic vehicles are low (Anderson, 2006:251).

One type of hypersonic vehicle capable of delivering the L/D ratios needed by the space plane has been labeled the waverider. As described by Anderson "the waverider is a supersonic or hypersonic vehicle that has an attached shock wave all along its leading edge (Anderson, 2006:251). By riding atop the shock wave the vehicle is able to keep the high pressure behind the shock wave beneath its "wing" in a more efficient manner. In fact, if the proper design for a space vehicle is chosen, coefficient of lift over coefficient of drag (C_l/C_d) values slightly greater than six can be attained (Anderson, 2006:251). It is because of the increased L/D ratios for a given angle of attack generated by a waverider that makes it the most likely design for space planes. Shown below is an example of a waverider design vehicle, the X-51A, as provided by the U.S. Air Force's information sheet (X-51A).

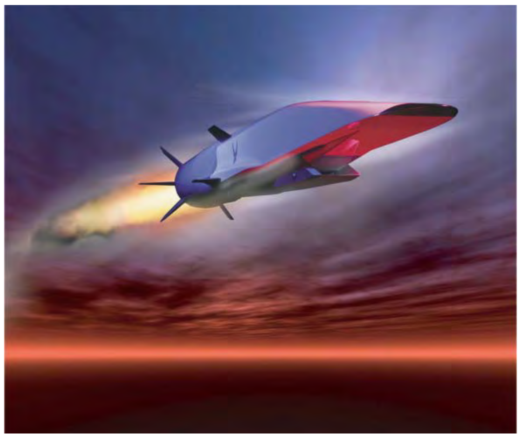


Figure 5: X-51A

Waveriders are typically referenced in many studies of aero-assisted vehicles. Many of these studies have focused on using waveriders not to change orbital parameters on its own, but to use an elliptical orbit and couple the waverider's lift forces with an onboard engine thrust as the vehicle dips into the atmosphere at perigee. Pienkowski and Jolley both conducted independent studies using this approach whose papers are described below.

Pienkowski proposed using the advancements in light weight lifting bodies over the past fifty years to develop a horizontal landing reusable spacecraft. This craft would be capable of changing its inclination by dropping into the atmosphere and firing a restart-able and throttle-able rocket engine. Utilizing data from the X-37 Pienkowski developed a simulation capable of a wide variety of candidate maneuvers and trajectories (Pienkowski, 2002:15). Using this model Pienkowski was able to vary the angles with which air hit the spacecraft and measure the resulting lift and drag forces. These values coupled with the thrust from the engine allowed Pienkowski to predict the resulting inclination changes of the spacecraft. Although much of his research focused on developing the control model for the engine; Pienkowski also developed a spacecraft model predicting inclination changes of 12 to 16 degrees within 10 orbits for light weight lifting bodies (Pienkowski, 2002:1).

Jolley looked at using a waverider to not only change the vehicle's inclination but also inclination changes to make the vehicle's orbit unpredictable. He used many of the same methods and equations of motion used by Pienkowski and assumed the vehicle would be in an elliptical orbit and use a thruster to maintain the orbit's perigee.

A graphical user interface was used to select different starting conditions (speed, angle of attack, orbital parameters) and propagate the orbit through a specified time window. Using this approach he was able to show an aero-assisted waverider significantly changed the arrival time over a target (Jolley, 2007:34). Figure 6 and Figure 7 on the next page taken from his work show the ground track changes and unpredictability an aero-assisted vehicle could provide against a satellite without aero-assistance.

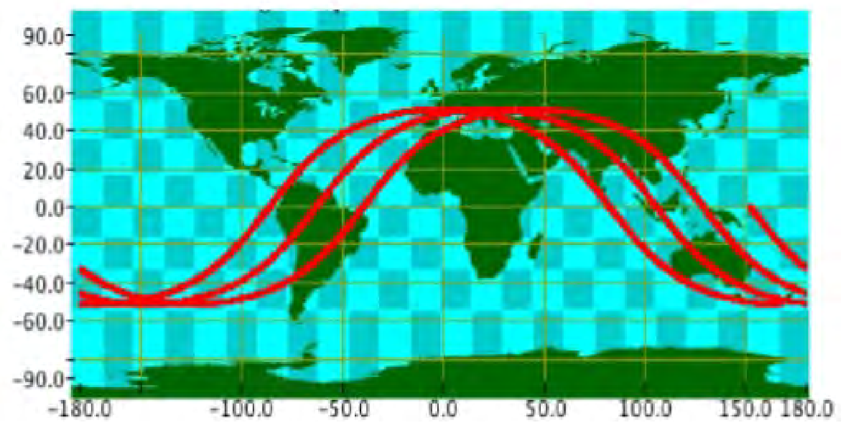


Figure 6: Fixed Orbit Trajectory

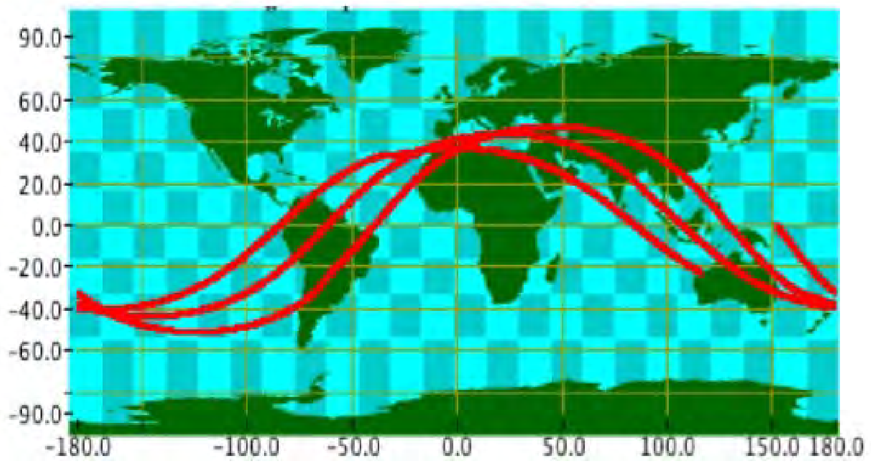


Figure 7: Aero-Assisted Orbit Trajectory

Jolley's work showed the vehicle could produce these maneuvers at a propulsion cost of 1/3 the amount of fuel a non aero-assisted vehicle would use. In addition he was able to show this maneuver could be done in as little as three orbits, greatly enhancing the unpredictability of the vehicle.

Although both of these approaches provided significant savings in velocity change for their respective missions, neither seemed to look into the effect of using a constant lift force in order to maintain desired orbit parameters. In order to find relevant data for a constant force approach a study in electric thrusters was conducted.

Electric Thruster/Constant Thrust Study

Electric thrusters have the capability of firing for extended periods of time and can provide the constant thrust motion this thesis looks to analyze. The drawbacks of these thrusters are their limited force, usually less than one Newton.

However, the equations developed to analyze their behavior for orbital propagation can be easily altered to include the effects of aerodynamic forces. Electric thrusters also provide an example for which this research can compare the added benefits from a space plane utilizing the aerodynamic forces. This thesis began by observing the work of Captain Timothy Hall.

Hall used examples of current and capabilities of projected future electric thrusters to model the effects of continuous thrusting for three scenarios: perigee height maintenance, temporal access improvement, and right ascension access improvement. Hall found improvements in two of the three areas, perigee height maintenance and temporal access. The research of this thesis will demonstrate the improved capabilities an aero-assisted vehicle can provide for perigee height maintenance by taking Hall's analysis a step further for the perigee maintenance. In his model, Hall used a burn from perigee to apogee to slowly increase the semi-major axis. Over time it was shown the orbit slowly increased in height but also circularized (Hall, 2010:22). The circular pattern would severely hamper the ability of a space plane to take advantage of aerodynamic forces since the satellite would be out of the range of atmospheric effects. The research of this thesis will show a burn can be conducted in a band across apogee so as to increase the semi-major axis without causing the orbit to circularize and allow the satellite to take advantage of the aerodynamic forces. A MatLab model will also be created to allow further manipulation of the thruster profile rather than relying on Satellite Tool Kit's propagator. A thorough vetting of the MatLab model will be conducted to insure it is within a reasonable margin of error when compared with results from Satellite Tool Kit.

Methodology

Chapter Overview

A foundation of understanding in orbital dynamics, Gaussian variation of parameters equations, orbital perturbations, and hypersonic aerodynamics is needed to develop models used here. This thesis will begin this discussion with an overview of the coordinate system used for the analysis.

Coordinate System

The coordinate system chosen for this thesis is the Gaussian coordinate system. The Gaussian coordinate system began as a means to provide an easier method for describing the relative motion between two satellites, a target and an interceptor; this is why it is often called the satellite coordinate system (Vallado, 2004:162). While this thesis will not study relative motion between satellites, the methods of using non-conservative forces incorporated with the Gaussian coordinate system will be utilized. The reason for this decision is the non-conservative nature of drag and lift.

There are two main variations of the Gaussian coordinate system according to Vallado. These systems are the RSW and NTW coordinate systems (Vallado, 2004:162). For the RSW coordinate system the R axis always points away from the center of the Earth, the S axis is in the direction of the satellite’s velocity vector but not necessarily parallel with it, and the W axis is normal to the orbital plane. The NTW system is similar to the RSW system however the satellites velocity vector is now used as the reference point. In other words the T axis is tangential to the orbit and always points to the velocity vector, the N axis is normal to the velocity vector and remains in the orbital plane, and the W axis is normal to the orbital plane as it was in the RSW system. This thesis will utilize the RSW frame. Figure 8 below gives a representation of the RSW frame.

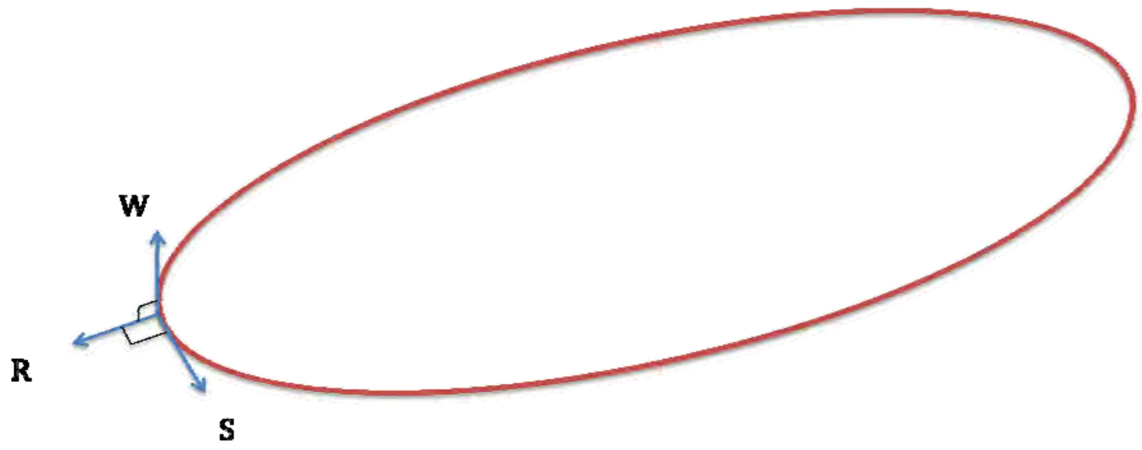


Figure 8: RSW Coordinate System

Gaussian VOP Equations

The Gaussian variations of parameters (VOP) equations are derived from the Lagrange VOP equations but include the effects of disturbing forces. These equations will be used extensively due to the continual change in the orbit from perturbing forces. Many of today's methods use a reference orbit to propagate forward with while the VOP continually updates the orbit as time progresses. This is extremely useful in allowing us to model the orbit from a wide variety of disturbing forces to include drag and lift. The derivations of these equations as derived by Bate, Mueller, and White are described below.

Gauss used the standard orbital elements a, e, i, Ω, ω, and M to derive his VOP equations. Bate, Mueller, and White used the RSW coordinate frame and provided a step-by-step derivation of Gauss's VOP equations which is shown below (Bate, 1971:397-406).

In the RSW coordinate system the perturbing force is

$$F = m(F_r\boldsymbol{R} + F_s\boldsymbol{F} + F_w\boldsymbol{W}) \tag{4}$$

and

$$\boldsymbol{r} = \boldsymbol{rR} \tag{5}$$

$$\boldsymbol{v} = \dot{r}\boldsymbol{R} + r\dot{\boldsymbol{v}}\boldsymbol{S} \tag{6}$$

Semi-major Axis

Using a per unit mass formulation the time rate-of-change of energy for a particular orbit can be expressed as

$$\frac{d\boldsymbol{\varepsilon}}{dt} = \frac{\boldsymbol{F}\cdot\boldsymbol{V}}{m} = \dot{\boldsymbol{v}}(\frac{dr}{d\boldsymbol{v}}F_r + rF_s) \tag{7}$$

and

$$\boldsymbol{\varepsilon} = -\frac{\mu}{2a} \tag{8}$$

Using these two equations the change in the semi-major axis over time can be expressed as

$$\frac{da}{dt} = \frac{da}{d\boldsymbol{\varepsilon}}\frac{d\boldsymbol{\varepsilon}}{dt} = \frac{\mu}{2\varepsilon^2}\frac{d\boldsymbol{\varepsilon}}{dt} \tag{9}$$

Using

$$\frac{dr}{d\boldsymbol{v}} = \frac{re\sin(\boldsymbol{v})}{1 + e\cos(\boldsymbol{v})} \tag{10}$$

$$h = r^2\dot{\boldsymbol{v}} = na^2\sqrt{1-e^2} \tag{11}$$

it can be shown the change in true anomaly over time is

$$\dot{\boldsymbol{v}} = \frac{na}{r^2}\sqrt{1-e^2} \tag{12}$$

Equations 6, 7 and 8 can be substituted into equation 9 to provide the final equation for the $\frac{da}{dt}$ equation.

$$\frac{da}{dt} = \frac{2e\sin(\boldsymbol{v})}{n\sqrt{1-e^2}}F_r + \frac{2a\sqrt{1-e^2}}{nr}F_s \tag{13}$$

Eccentricity

For the following elements the time rate of change of angular momentum is needed. This rate can be expressed as the moment of perturbing forces acting on the system resulting in the equation below.

$$\frac{d\boldsymbol{h}}{dt} = \frac{1}{m}(\boldsymbol{r} \times \boldsymbol{F}) = rF_s\boldsymbol{W} - rF_w\boldsymbol{S} \tag{14}$$

The angular momentum for an orbit can also be expressed as the change in length in the **W** direction and the transverse component along the plane of rotation for the following equation.

$$\frac{d\boldsymbol{h}}{dt} = \dot{h}\boldsymbol{W} + h\frac{d\alpha}{dt}\boldsymbol{S} \tag{15}$$

By comparing components of equations 14 and 15 the time rate of change for the magnitude of the angular momentum is

$$\frac{dh}{dt} = rF_s \tag{16}$$

Using $p = a(1-e^2)$ an expression for eccentricity can be found.

$$e = (1 - \frac{p}{a})^{1/2} = (1 - \frac{h^2}{\mu a})^{1/2} \tag{17}$$

The time rate of change of eccentricity is found to be

$$\frac{de}{dt} = -\frac{h}{2\mu ae}\left(2\frac{dh}{dt} - \frac{h}{a}\frac{da}{dt}\right) \tag{18}$$

and substituting dh/dt and da/dt the expression can be further simplified to

$$\frac{de}{dt} = \frac{\sqrt{1-e^2}\sin(\boldsymbol{v})}{na}F_r + \frac{\sqrt{1-e^2}}{na^2e}\left[\frac{a^2(1-e^2)}{r} - r\right]F_s \tag{19}$$

Inclination

Using the dot product definition of inclination and differentiating provides the following

$$-sin(i)\frac{di}{dt} = \frac{h\left(\frac{d\boldsymbol{h}}{dt}\cdot\boldsymbol{K}\right) - (\boldsymbol{h}\cdot\boldsymbol{K})\frac{dh}{dt}}{h^2} \tag{20}$$

$$= \frac{h(rF_s\boldsymbol{W} - rF_w\boldsymbol{S})\cdot\boldsymbol{K} - h\,cos(i)rF_s}{h^2}$$

These equations can be further simplified using

$$\boldsymbol{W}\cdot\boldsymbol{K} = cos(i) \tag{21}$$

$$\boldsymbol{S}\cdot\boldsymbol{K} = sin(i)cos(u) \tag{22}$$

where u is argument of latitude and is equal to ω + ν. Substituting these equations into equation 20 provides the further simplified equation

$$\frac{di}{dt} = \frac{rF_w cos(u)}{na^2\sqrt{1-e^2}} \tag{23}$$

Right Ascension of the Ascending Node

Once again we begin with the dot product definition of Ω and differentiate to obtain

$$-sin(\Omega)\frac{d\Omega}{dt} \tag{24}$$

$$= \frac{\boldsymbol{I}\cdot\left(\boldsymbol{K}\times\frac{d\boldsymbol{h}}{dt}\right)|\boldsymbol{K}\times\boldsymbol{h}| - \boldsymbol{I}\cdot(\boldsymbol{K}\times\boldsymbol{h})\frac{d}{dt}|\boldsymbol{K}\times\boldsymbol{h}|}{|\boldsymbol{K}\times\boldsymbol{h}|^2}$$

$$= \Big\{\boldsymbol{I}\cdot[\boldsymbol{K}\times(rF_s\boldsymbol{W} - rF_w\boldsymbol{S})]\, hsin(i)$$

$$- h\,cos(\Omega)sin(i)\left(\frac{dh}{dt}sin(i) + h\,cos(i)\frac{di}{dt}\right)\Big\}\frac{1}{h^2 sin^2(i)}$$

Using the dot product relations

$$\boldsymbol{I}\cdot\boldsymbol{K}\times\boldsymbol{W} = cos(\Omega)sin(i) \tag{25}$$

$$\boldsymbol{I}\cdot\boldsymbol{K}\times\boldsymbol{S} = \boldsymbol{I}\times\boldsymbol{K}\cdot\boldsymbol{S} = -\boldsymbol{J}\cdot\boldsymbol{S} \tag{26}$$

$$= sin(\Omega)sin(u) - cos(\Omega)cos(u)cos(i)$$

We can now use the definitions above, $\frac{di}{dt}$, and $\frac{dh}{dt}$ to further simplify the equation

$$\frac{d\Omega}{dt} = \frac{rF_w sin(u)}{h\,sin(i)} = \frac{rF_w sin(u)}{na^2\sqrt{1-e^2}\,sin(i)} \tag{27}$$

Argument of Perigee

Using the dot product definition for the argument of latitude and substituting ω + ν in for u we get the following equation.

$$\frac{(\boldsymbol{K}\times\boldsymbol{h})\cdot\boldsymbol{r}}{|\boldsymbol{K}\times\boldsymbol{h}|} = rcos(\omega+\nu) \tag{28}$$

We can then differentiate and $\frac{d\omega}{dt}$ over to get

$$\frac{d\omega}{dt} \tag{29}$$

$$= \left\{\frac{1}{h^2 sin^2(i)\, r\, sin(u)}\right\}\left\{-h\, sin(i)[\boldsymbol{K} \times (rF_S\boldsymbol{W} - rF_W\boldsymbol{S}) \cdot \boldsymbol{r}]\right.$$

$$+ (\boldsymbol{K} \times \boldsymbol{h} \cdot \boldsymbol{r})\left[\frac{dh}{dt} sin(i) + h\, cos(i)\frac{di}{dt}\right]$$

$$\left. - \frac{d\nu}{dt} h^2\, r\, sin(i) sin(u)\right\}$$

where

$$\boldsymbol{K} \times \boldsymbol{W} \cdot \boldsymbol{r} = r\, sin(i) cos(u) \tag{30}$$

$$\boldsymbol{K} \times \boldsymbol{S} \cdot \boldsymbol{r} = r\boldsymbol{S} \times \boldsymbol{R} \cdot \boldsymbol{K} = -r\boldsymbol{W} \cdot \boldsymbol{K} = -r\, cos(i) \tag{31}$$

$$\boldsymbol{K} \times \boldsymbol{h} \cdot \boldsymbol{r} = r\, h\, sin(i) cos(u) \tag{32}$$

The only term that we have not derived an expression for is $\frac{d\nu}{dt}$. We will do this by using the conic equation

$$r\left(1 + e\cos(\nu) = \frac{h^2}{\mu}\right) \tag{33}$$

and differentiate the terms only affected by perturbations

$$r\left(\frac{de}{dt}\cos(\nu) - e\sin(\nu)\frac{d\nu}{dt}\right) = \frac{dh}{\mu}\frac{dh}{dt} \tag{34}$$

$$r\, e\, sin(\nu)\frac{d\nu}{dt} = r\, cos(\nu)\frac{de}{dt} - \frac{2h}{\mu}\frac{dh}{dt} \tag{35}$$

To simplify the algebra Bate, Mueller, and White used the identity and its derivative

$$\mu\, r\, e\, sin(\nu) = h\, \boldsymbol{r} \cdot \mathbf{v} \tag{36}$$

$$\mu\, r\, \frac{de}{dt}\sin(\nu) + \mu\, r\, e\cos(\nu)\frac{d\nu}{dt} = \frac{dh}{dt}\boldsymbol{r} \cdot \mathbf{v} + h\boldsymbol{r} \cdot \dot{\boldsymbol{v}} \tag{37}$$

If we multiply equation 35 by $\mu\sin(\nu)$ and 37 by $\cos(\nu)$ and add them together the following equation is found

$$\frac{d\nu}{dt} = \frac{1}{r\ e\ h}\left[\mathrm{p}cos(\nu)\boldsymbol{r}\cdot\dot{\mathbf{v}} - (\mathrm{p}+r)\sin(\nu)\frac{dh}{dt}\right] \tag{38}$$

We can then substitute into equation 29 to get the following equation for $\frac{d\omega}{dt}$ separated out according to the three components

$$\frac{d\omega}{dt} = \left(\frac{d\omega}{dt}\right)_r + \left(\frac{d\omega}{dt}\right)_s + \left(\frac{d\omega}{dt}\right)_w \tag{39}$$

$$\left(\frac{d\omega}{dt}\right)_r = -\frac{\sqrt{1-e^2}\cos(\nu)}{n\ a\ e}F_r \tag{40}$$

$$\left(\frac{d\omega}{dt}\right)_s = \frac{\mathrm{p}}{e\ h}\left[\sin(\nu)\left(1+\frac{1}{1+e\cos(\nu)}\right)\right]F_s \tag{41}$$

$$\left(\frac{d\omega}{dt}\right)_w = -\frac{r\cot(i)\sin(u)}{n\ a^2\sqrt{1-e^2}}F_w \tag{42}$$

Mean Anomaly

Beginning with

$$M_0 = E - e\sin(E) - n(t-t_0) \tag{43}$$

$$\cos(\nu) = \frac{\cos(E)-e}{1-e\cos(E)} \tag{44}$$

$$\sin(\nu) = \frac{\sqrt{1-e^2}\sin(E)}{1-e\cos(E)} \tag{45}$$

and differentiating M_0

$$\frac{dM_0}{dt} = \frac{dE}{dt} - \frac{de}{dt}\sin(E) - e\cos(E)\frac{dE}{dt} - \frac{dn}{dt}(t-t_0) \tag{46}$$

one can substitute in to get the final equation for $\frac{dM_0}{dt}$ for elliptical orbits only

$$\frac{dM_0}{dt} = -\frac{1}{n\,a}\left(\frac{2r}{a} - \frac{1-e^2}{e}\cos(\nu)\right)F_r - \frac{(1-e^2)}{n\,a\,e}\left(1 + \frac{r}{a(1-e^2)}\right)\sin(\nu)\,F_s - t\,\frac{dn}{dt} \tag{47}$$

Perturbations

Perturbations are the main complexity in accurately propagating an orbit. There are many perturbations to consider when accurately predicting a spacecraft's orbit; however this paper will focus only on drag and lift. The third most dominant force due to the J2 perturbation will be ignored along with all others due to its relative small size when compared with the drag, lift, and thruster forces. A brief description of the calculations used to find the J perturbations is given in the next section along with Table 1 which shows the relative magnitudes of the forces, the reason for ignoring all perturbations other than drag and lift.

J Perturbations

The J perturbations accounts for the Earth being asymmetrical and is modeled using the potential function used by Bate, Mueller, and White,

$$\Phi = \frac{\mu}{r}\left[1 - \sum_{n=2}^{\infty} J_n \left(\frac{r_e}{r}\right)^n P_n sin(L)\right] \tag{48}$$

where μ = the gravitational parameter
r = the position vector of the satellite
J_n = a coefficient determined by experimental observation
r_e = the radius of the Earth at the equator
P_n = the Legendre polynomial

L = the geocentric latitude, $\sin(L) = \frac{z}{r}$

The acceleration can then be calculated by finding the gradient of the potential function which is shown below.

$$\boldsymbol{a} = \nabla = \frac{\delta}{\delta x}\boldsymbol{I} + \frac{\delta}{\delta y}\boldsymbol{J} + \frac{\delta}{\delta z}\boldsymbol{K} \tag{49}$$

Taking the partial derivative of this function supplies the necessary accelerations for calculating the perturbations due to the Earth being asymmetrical. The equations for nodal regression and perigee rotation rates are shown below.

$$\dot{\boldsymbol{\Omega}} = -\frac{\boldsymbol{3}}{\boldsymbol{2}}\frac{\boldsymbol{nR_{\oplus}^{2}}}{\left(\boldsymbol{a}(\boldsymbol{1}-\boldsymbol{e^{2}})\right)^{2}}\boldsymbol{J_{2}cos}(\boldsymbol{i}) \tag{50}$$

$$\dot{\boldsymbol{\omega}} = -\frac{\boldsymbol{3}}{\boldsymbol{4}}\frac{\boldsymbol{nR_{\oplus}^{2}}}{\left(\boldsymbol{a}(\boldsymbol{1}-\boldsymbol{e^{2}})\right)^{2}}\boldsymbol{J_{2}}\{\boldsymbol{4}-\boldsymbol{5sin^{2}}(\boldsymbol{i})\} \tag{51}$$

Since both equations depend on the sine of inclination, an inclination of 0 degrees will be used to determine the maximum possible regression and rotation rate.

Drag

Drag will be calculated using the simple drag equation, Equation 2. An area of 2.67 m^2, a coefficient of drag of 1, an air density found in the U.S. Standard Atmospheric model for the respective heights, and the corresponding speed based on the eccentricity will be used. The corresponding speed can be found by Equation 52 below.

$$V^{R} = V_{Orbital} - V_{atmosphere} \tag{52}$$

The relative velocity of the spacecraft is found by approximating the speed of the atmosphere itself as angular motion of the Earth multiplied by the corresponding radius component. By subtracting this value from the velocity component found using equation

52 the relative velocity for use in the atmospheric drag calculation can be found.

As stated earlier, Table 1 provides the comparison of the J perturbation forces for Ω and ω at their maximum against the forces of drag for two different radiuses of perigee. Please note the drags are shown for the exact second at perigee while the regression and rotation are shown for the accumulation of an entire day. Drag accumulation for one orbit will actually be much larger but the differences are made clear with this simpler analysis. Since the analysis of this thesis will use radius of perigees below 6500 km the decision was made to neglect central body and all other perturbations.

Table 1: Drag and Central Body Perturbation Comparisons

Radius of Perigee (km)	**Eccentricity**	**Drag (N) *One Second**	**J2 Right Ascension (Degrees) *One Day**	**J2 Argument of Perigee (Degrees) *One Day**
6500	0.01	1.29081	0.157164	0.314328
	0.1	1.41334	0.114848	0.229695
	0.25	1.61824	0.067657	0.135315
	0.5	1.96125	0.025575	0.05115
	0.75	2.30576	0.006643	0.013286
6600	0.01	0.00925	0.148986	0.297973
	0.1	0.01013	0.108872	0.217743
	0.25	0.01160	0.064137	0.128274
	0.5	0.01407	0.024244	0.048489
	0.75	0.01654	0.006298	0.012595

Lift

Lift was found in the same manner as drag and the equations are identical except for replacing the C_D with the C_L and the assumption of the same reference area. The equation is shown below.

$$F_L = \frac{1}{2}\rho C_L A V_R^2 \tag{53}$$

The Atmosphere

The Earth's atmosphere is the main variable in which scientists do not have a firm model for drag analysis. There are just too many factors to accurately predict the exact density of the atmosphere. Because of these complexities there is great uncertainty in any atmospheric model. However, because of this uncertainty many studies have been conducted in an attempt to fully understand the atmosphere. From these studies, an abundant amount of data has allowed scientists to create models that model the nominal behavior of the atmosphere within a reasonable percentage of error. One such model is the U.S. Standard Atmosphere Model, 1976.

The U.S. Standard Atmosphere Model, 1976 stated goal is to support the emerging missile industry with a description of the atmosphere beyond current operating altitudes of conventional aircraft (NASA, 1976: xiii). It is because of the amount of data and mid-latitude focused concentration of the atmosphere this model uses that it was chosen for the atmospheric model used by this paper. Vallado also used this model in many of his studies and has produced an algorithm for calculating the density of the atmosphere for a given satellite height (Vallado, 525). MatLab code based on this algorithm was also readily available to the public leading to the second factor for the US Standard Atmosphere Model 1976 model choice, ease of integration into the overall code. A complete copy of the code can be found in Appendix B with the rest of the MatLab code used by this research and is titled atmos76.m.

Coefficient of Drag and Lift

The coefficients of drag and lift are numbers based on a particular Reynold's number, viscosity, and angle of attack. There are many methods for finding these values, however the degree of complexity and calculations needed distract from the purpose of this thesis. For this reason a coefficient of drag value of one will be used for all simulations. The defense of this choice is given in the following paragraphs.

Most satellite models utilize a modified Newtonian flow for a flat plate to determine the drag coefficient.

$$C_{p_{max}} = \frac{2}{\gamma M_\infty^2}\left\{\left[\frac{(\gamma+1)^2 M_\infty^2}{4\gamma M_\infty^2 - 2(\gamma-1)}\right]^{\gamma/(\gamma-1)}\left[\frac{1-\gamma+2\gamma M_\infty^2}{\gamma+1}\right] - 1\right\} \qquad (54)$$

where γ is the ratio of specific heats
M_∞ is the free stream Mach number

Figure 9 shows the value of $C_{p_{max}}$ for various free stream Mach numbers and ratios of specific heat, γ, conditions.

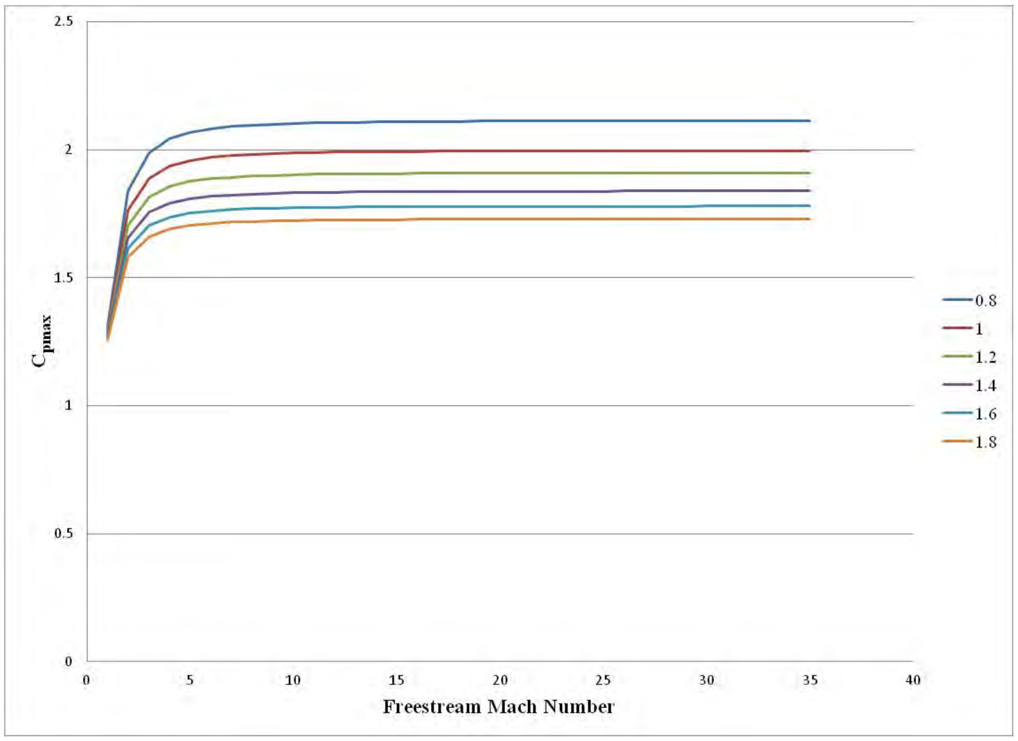


Figure 9: Variation of C_{pmax} with M_∞ and γ

Since γ will tend towards the lower end of the graph and most satellites fly through the atmosphere with a shape roughly that of a flat plate, most satellites use the $C_{p_{max}}$ approach. It is also important to note the basic Newtonian flow theory can also be used to predict the coefficient of drag for a sphere and cylinder with an infinite span at 1 and 4/3 respectively. These shapes are more aerodynamically efficient in high speed flows and produce smaller coefficients of drag. Because of this these coefficient of drag values will be more applicable to the design of this thesis's space plane coefficient of drag estimate.

In addition to the estimation methods described above, Anderson has documented the coefficient of drag versus Mach number for STS-5. This chart is shown below in Figure 10 (Anderson, 2006:87).

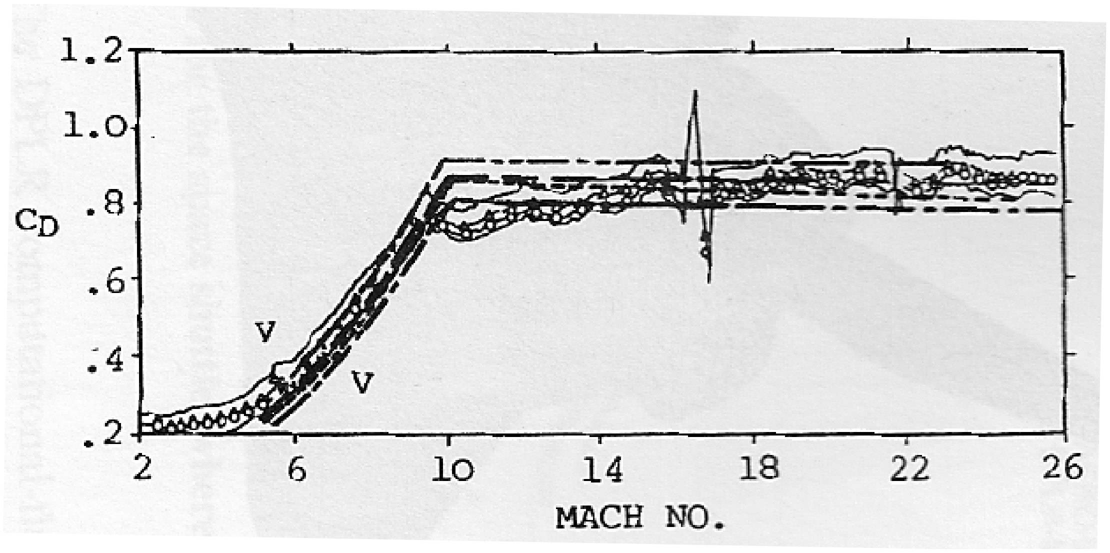


Figure 10: STS-5 C_D vs. Mach Number

Based on this data it appears the shuttle has a coefficient of drag around 0.8 for most of its flight around high mach numbers. It is safe to assume a waverider could provide better drag coefficients or at the very least match those of the shuttle.

Reviewing all of this data it appears the maximum drag coefficients expected for any shape would be around two. However, these numbers are produced by satellite designs, which could only be mimicked by the waverider if it oriented its bottom side perpendicular to the velocity vector. This scenario is unlikely. The more aerodynamically shaped sphere and cylinder produce coefficients near one. Again, these shapes are not likely to be incorporated by the waverider. The final piece of data provides a coefficient of drag of 0.8 for the space shuttle at high mach numbers. The design of the shuttle would be the most likely candidate to resemble the final design of waveriders. Based on this data it is safe to assume the coefficient of drag could at least match that of the shuttles'.

Because of this similarity and the sphere and cone coefficients being near 1, a decision to choose 1 as the model's coefficient of drag was made. This number is close to that of the shuttle's, but incorporates an error bias since no on orbit data is available for waveriders. In practice the coefficient of drag should be less than 1 and thus be able to outperform the model used in this thesis. However, the trending will remain the same and the reader should be able to estimate what an improved coefficient of drag would provide based on the results section.

Using this base coefficient of drag of 1 and lift to drag ratios of 0.5, 1, and 1.5 provided the most relevant data and would be in line with expected values as stated earlier. Choosing these values rather than calculating coefficients within the program is based on computation time and simplification of the problem. The amount of computation time needed to effectively calculate drag and lift coefficients would have greatly increased computation time and complexity of the problem for little to no gain in understanding the trends associated with it.

Once the lift and drag accelerations are added together, the numerical method described below can be used to find the new position and velocity vector of the satellite for a pre-selected change in time. This process is then repeated over a specified length in time for the orbit.

Numerical Integrator

The numerical integrator chosen for this paper is ODE45, which is found in MatLab. This integrator is based on an eight step Runge-Kutta method and is highly accurate and adaptable. ODE45 will be utilized by finding new values for the classical orbital elements (COEs) after a specified time step based on initial COEs and the Lagrange VOP equations. The maximum time step chosen for this research is five seconds for any ODE45 calculation.

Computer Model

By using the equations and integrator discussed above, various scripts can be created to perform the calculations and propagate the orbit forward. These scripts were then implemented into a master script for the final program. This master script used initial classical orbital elements along with initial thrust profiles and outputted a ground track for the satellite, a coverage list of when the satellite has line of sight on a specified target, and plots for the classical orbital elements and forces versus time. For a complete list of the scripts and the scripts themselves please see Appendix B. Figure 11 provides a flowchart of how the various files are implemented into the master script.

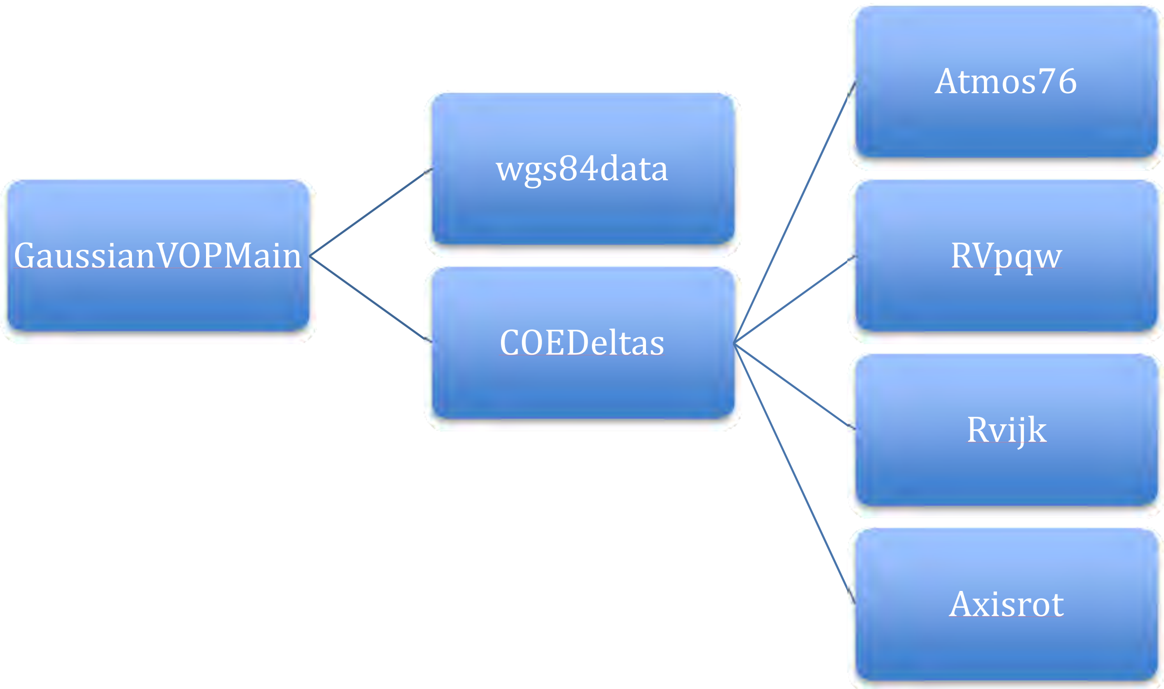


Figure 11: Computer Code Structure

The code utilizes a max step size of 30 seconds and is useful for orbits of eccentricities greater than 0.001 and inclinations greater than 5 degrees. This step size is not to be confused with the maximum step size chosen for the ODE45 MatLab function. In scenarios involving eccentricities less than 0.001 and inclinations less than five degrees a different set of equations would be needed. This need for new equations is due to the singularities experienced by the Gaussian VOP equations for eccentricity and inclination values close to 0. However, based on the scenarios chosen for this research these equations will suffice.

Results and Discussion

Chapter Overview

This chapter will provide the results of the simulations for various Lift vs. Drag coefficients and thrusting/non-thrusting satellite profiles. The simulations will use the methodology described in the previous chapter along with any needed inputs described within this chapter as well. The results will also be discussed within this chapter to provide the reader with a better understanding of the analysis.

Verification and Validation

In order to successfully utilize the code described above a verification and validation process was needed. The first step in this process involved using Satellite Tool Kit (STK) to verify the outputs of the code were correct. Both the code described above and STK were given a satellite with the following COEs and epoch time.

Table 2: Initial COEs

a	6648.137
e	0.005
i	40 deg
Ω	0 deg
ω	0 deg
M	0 deg
Epoch Time	3 June 2011 12:00

The satellite was then propagated forward in time utilizing a two-body approach without perturbations for four orbits and the results were compared together. Based on the results, the first five COEs did not change, as was expected, and the mean anomaly for the code was slightly above 0.002 percent of the mean anomaly for STK for each perigee crossing. This percentage is well within acceptable limits and is most likely caused by a slight timing error. Table 3 provides the side-by-side comparisons for the code and STK mean anomaly outputs at each perigee crossing based on the original orbital period.

Table 3: Mean Anomaly Perigee Crossing Values

Perigee Crossing	MatLab Code Mean Anomaly	STK Mean Anomaly	Percent Difference
1st	6.283323	6.283185	0.002196
2nd	12.56665	12.56637	0.002228
3rd	18.84997	18.84956	0.002175
4th	25.13329	25.13274	0.002188

The second step in validating the code was to conduct the same study as before but also include drag. The specifications chosen for the space plane design incorporating drag are shown in table 4 on the next page. Semi-major Axis was decreased to 7197 km and eccentricity was increased to 0.1 to provide for better comparison with values used later in this thesis.

Table 4: Space Plane Design Specifications

Parameter	Value
Area (m^2)	2.67
Coefficient of Drag	1.0
Mass (kg)	2000
Flight Path Angle (degrees)	2

The propagators were again run for four orbits and the COEs for both the code and STK at each perigee crossing based on the original orbital period are shown below for comparison.

Table 5: STK & MatLab Code Drag Comparisons

Perigee Crossing	Propagator	a (km)	e	*i* (rad)	Ω (rad)	ω (rad)	M (rad)
1st	STK	7193.079	0.099396	0.698132	0	0.000418	6.283185
	MatLab	7192.589	0.099334	0.698132	0	0.000366	6.283185
	Percent Difference	0.006825	0.062449	0	0	12.34743	0
2nd	STK	7177.4	0.097435	0.698132	0	0.000385	12.56637
	MatLab	7183.092	0.098149	0.698132	0	0.000159	12.56637
	Percent Difference	0.07931	0.733134	0	0	58.60008	0
3rd	STK	7173.607	0.096961	0.698132	0	0.000402	18.84956
	MatLab	7171.909	0.096748	0.698132	0	0.001016	18.84956
	Percent Difference	0.023665	0.219957	0	0	153.0352	0
4th	STK	7163.789	0.095729	0.698132	0	0.000454	25.13274
	MatLab	7159.936	0.095242	0.698132	0	0.001914	25.13274
	Percent Difference	0.053775	0.508854	0	0	321.853	0

For this reason the MatLab and STK simulations were propagated out to the point both codes predicted re-entry into Earth's atmosphere. This correlates to a time on orbit of 2 days, 17 hours, and 55 minutes. The STK had a predicted termination point of 2 days 23 hours and 10 minutes. This is approximately 5 and 1/2 hours later than the MatLab code predicted. Table 6 shows the COEs and percent differences for a time of 23,000 seconds. This value was chosen for this particular scenario because it is approximately one orbit before the time when the satellite is expected to re-enter the atmosphere as predicted by the MatLab code. The COEs were taken one orbit before termination in order to limit the amount of noise the large drag forces would cause at re-entry.

Table 6: STK & MatLab Code Drag Comparisons 40th Crossing

Propagator	a (km)	e	*i* (rad)	Ω (rad)	ω (rad)	M (rad)
STK	6666.4	0.028784	0.696823	0	6.2787	1.304878
MatLab	6569.593	0.014679	0.698132	0	0.214585	2.562427
Percent Difference	1.452173	49.00404	0.187852	0	96.58234	96.37292

Based on the data the inclination and right ascension did not change as was expected since there were no forces perpendicular to the orbital plane. The mean anomaly and argument of perigee both had large percent differences, but these values only predict where the satellite is in a particular orbit. Because this thesis is analyzing the time on orbit (affected by semi-major axis and eccentricity) and not the location of the orbit the author decided to accept the large percent differences for these COEs. Based on the values for semi-major Axis and eccentricity it appears the STK model is producing a slightly smaller specific force due to drag than the MatLab code which is resulting in a longer time on orbit. This can also be shown by Table 5 as the changes in argument of perigee are slightly smaller when compared with the MatLab predictions. The semi-major axis only had a 1.5 percent difference and is acceptable. The eccentricity produced a percent difference of 50% which would not be acceptable if this thesis was not focused on simple trending. Since the focus of this thesis is trending the effects lift and drag have on a satellite and because the codes were producing similar models with the difference being position of the satellite and predicted re-entry a determination to proceed with the

MatLab model was made.

Scenario Analysis

In order to begin the analysis, the first step chosen was to create various plots based on the satellite's true anomaly and a set of chosen semi-major axis, eccentricity, specific force, and inclination values using the VOP equations. The specific force and inclination were set at 1 Newton and 40 degrees respectively. The different sets of semi-major axis and eccentricity are tabulated below.

Table 7: Semi-major Axis and Eccentricity Values

a	e
6800	0.0001
"	0.001
"	0.01
"	0.1
7000	0.0001
"	0.001
"	0.01
"	0.1
10000	0.0001
"	0.001
"	0.01
"	0.1
20000	0.0001
"	0.001
"	0.01
"	0.1

Using these sets, various charts were created with the specific COE's time derivative on the y-axis and true anomaly on the x-axis. A small subset of these graphs is displayed below; for a complete listing of all the graphs please see Appendix C. Each chart shows how a specified COE's rate of change (y-axis) is affected by the satellite's true anomaly (x-axis). The various COE rates of change are shown with the semi-major axis set at 6,800; 7,000; 10,000; and 20,000 km for a particular eccentricity or the rates are shown with varying eccentricities at a particular semi-major axis. These variations are shown in the right-hand legend.

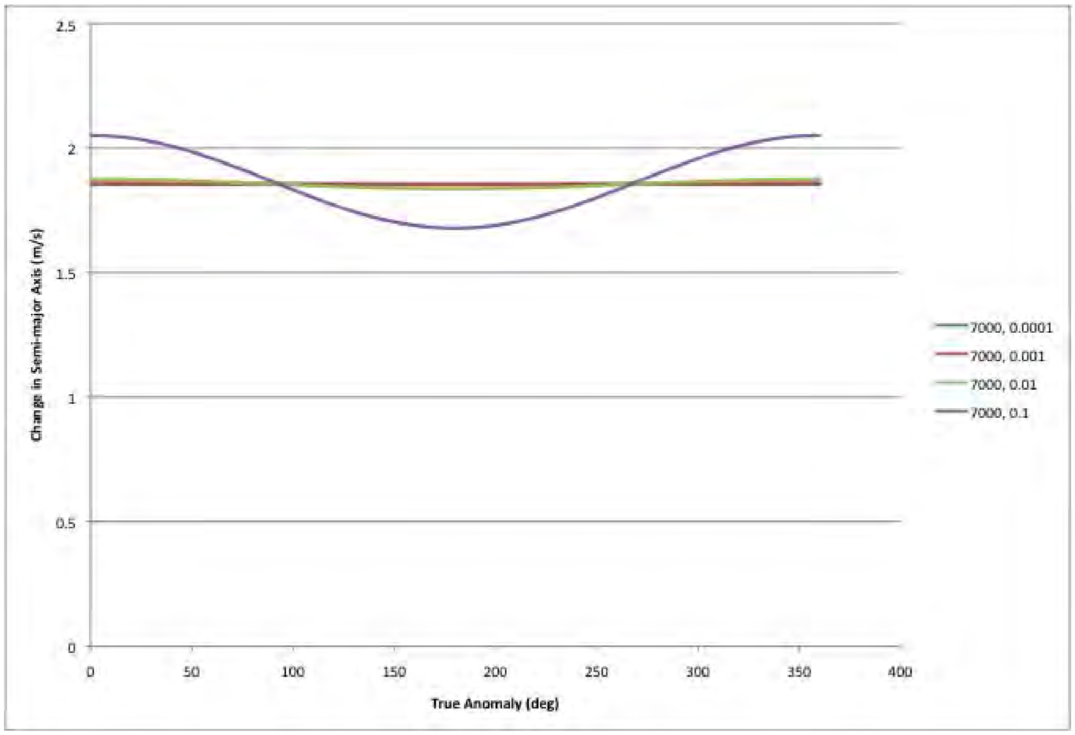


Figure 12: Rate of Change in Semi-major Axis Due to Changing Eccentricity and 7000 km Semi-major Axis, S-direction

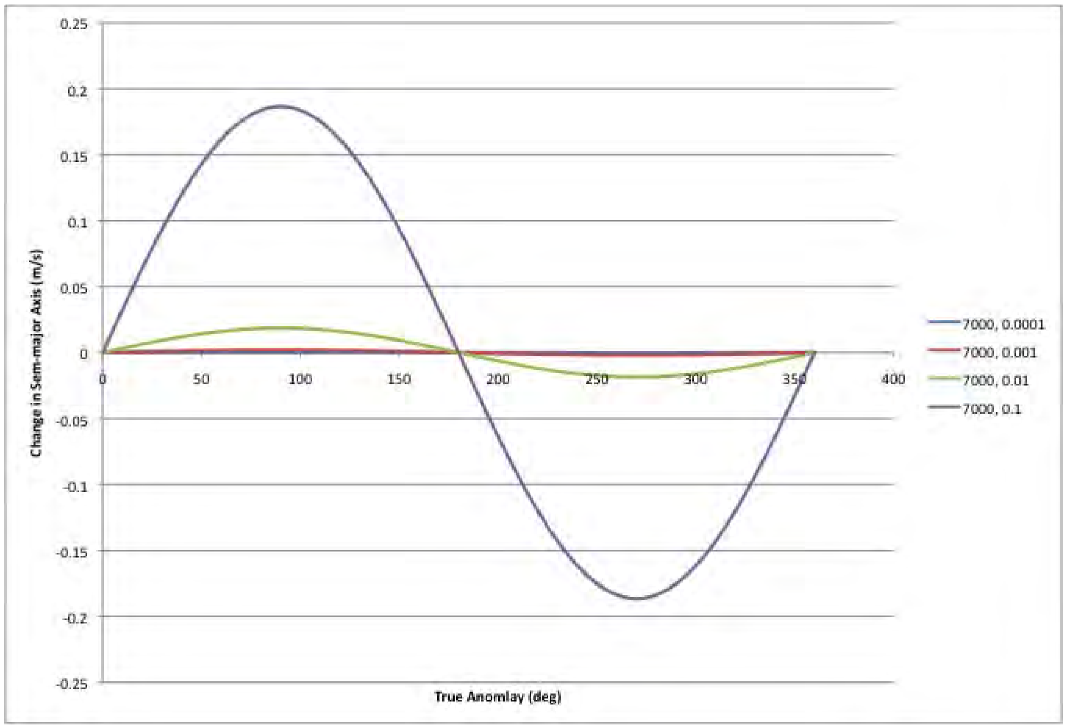


Figure 13: Rage of Change in Semi-major Axis Due to Changing Eccentricity and 7000 km Semi-major Axis, R-direction

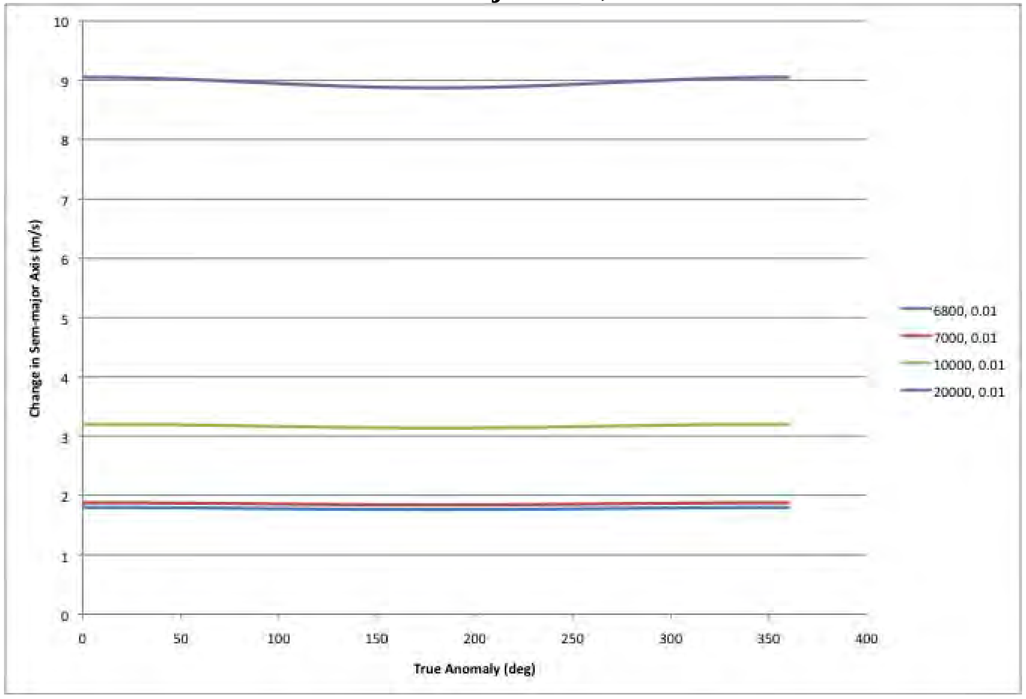


Figure 14: Rate of Change in Semi-major Axis Due to Changing Semi-major Axis and 0.01 Eccentricity, R-direction

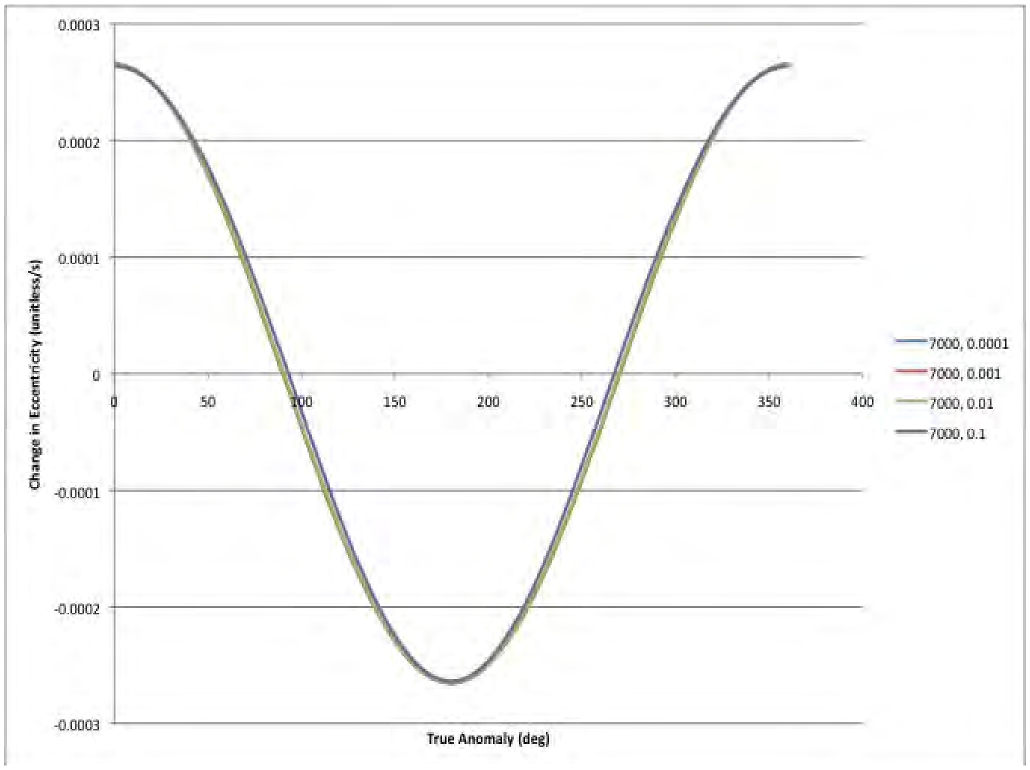


Figure 15: Rate of Change in Eccentricity Due to Changing Eccentricity and 7000 km Semi-major Axis, S-direction

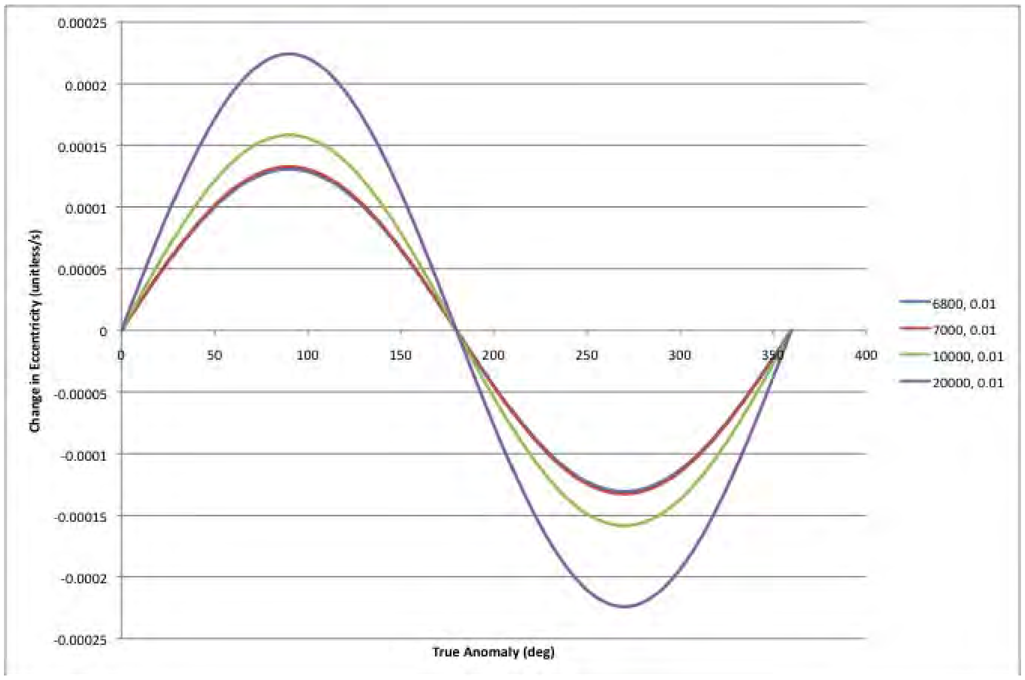


Figure 16: Rate of Change in Eccentricity Due to Changing Semi-major Axis and 0.01 Eccentricity, R-direction

These graphs shown above show the change in either semi-major axis or eccentricity at a particular instant for a force in the *S* or *R* direction. It is important to note that if a satellite has a larger semi-major axis then a force in either the *S* or *R* direction will cause a larger instantaneous change for both the semi-major axis and eccentricity. A satellite with a larger eccentricity will only create a more cosine curve effect for semi-major axis and have a little effect for eccentricity for burns in both the *S* and *R* direction. Right ascension of the ascending node, argument of perigee, inclination, and mean anomaly at epoch all have a cyclic value that depends on true anomaly for the burns in either the *S*, *R*, or *W* direction. The reader is encouraged to review these graphs in Appendix C in order to familiarize themselves with the expected time rate of change values for the different COEs. This will also allow the reader to better understand which forces affect which COE.

A brief summary of the affect each force has on the different COEs is shown in Table 8 on the next page. Utilizing the charts in Appendix C and assuming positive forces shows the direction of change (positive, negative) for each variable at true anomaly angles of 0, 90, 180, and 270 degrees.

Table 8: COE Direction of Change for a Positive Force

Variable/ Force Direction	True Anomaly			
	0	90	180	270
Force S				
a	+	+	+	+
e	+	0	-	0
ω	0	+	0	-
M_0	0	-	0	+
Force R				
a	0	+	0	-
e	0	+	0	-
ω	-	0	+	0
M_0	0	+	0	-
Force W				
i	+	0	-	0
Ω	0	+	0	-
ω	0	-	0	+

At first glance it may seem this table provides a simple means of choosing the necessary forces to transfer a satellite into another orbit. For example, providing a force in the positive s or r direction would cause an increase in orbit size. However, due to the coupling of the equations a force in the positive s direction would also change the satellite's eccentricity, argument of perigee, and epoch mean anomaly in either a positive or negative direction depending on the true anomaly. While this may not be a problem, the initial goal of increasing the semi-major axis has some unintended consequences.

A perfect example of this is the work done by Hall described earlier. Hall used a sample of Hall thrusters to produce a force oriented in the positive *S* direction which only fired while traveling from perigee to apogee to raise the spacecraft's orbit in order to maintain a near circular orbit (Hall, 2010:22). This orbit maneuver capability is due to the coupling of the equations. If Figure 12 and Figure 15 above are reviewed, this burn from perigee to apogee in the *S* direction would cause the semi-major axis to increase and the eccentricity time rate of change would increase part way and decrease the rest. For the purpose of this demonstration the changes in the argument of perigee and epoch mean anomaly will be neglected.

At first glance it appears the eccentricity should have a net effect of 0 due to the cosine nature of the eccentricity change. However; since the eccentricity time rate of change is dependent on the semi-major axis as well, and the fact there was a net increase of semi-major axis over the course of the burn, a negative net change in eccentricity results. This is due to the increase portion of the eccentricity cosine curve having a larger semi-major axis value over the course of the burn when compared with the decrease portion.

By burning from perigee to apogee the satellite is able to raise its orbit while maintaining a circular orbit. An important note is the exact opposite would occur if the burn was conducted with a significant portion in the 270 degrees to 90 degrees band. The satellite would increase its semi-major axis as well as its eccentricity due to the increased burn time near perigee.

Maintaining Radius of Perigee

Current methods for increasing the semi-major axis and maintain a perigee radius use a burn in the *S* direction only. These burns are typically done by large boost motors and are one time use devices. The analysis in this section will look at using a constant force thruster coupled with aerodynamic forces to maintain a relatively constant perigee while traveling through the atmosphere.

Using Figure 12 and Figure 13 it can be shown a specific force in the *S* direction more than doubles the gain of semi-major axis for any given eccentricity when compared with the *R* direction. It can also be shown for low eccentricities the curve reduces to a straight line with its magnitude being close to two and zero for the *S* and *R* directions respectively. For this reason most burns to increase the semi-major axis are usually in the *S* direction and this thesis will follow suit.

The analysis will look at holding the altitude of perigee as close to 80 and 100 km as possible, both perigee points will be analyzed at multiple eccentricities. Based on these criteria the COEs for each simulation run are listed below. These COEs will be used throughout the analysis of semi-major axis.

Table 9: List of COEs for Altitude of Perigee = 100 km

a (km)	e	*i* (deg)	Ω (deg)	ω (deg)	M (deg)
6510.69	0.005	40	0	0	180
6527.09	0.0075	40	0	0	180
6543.57	0.01	40	0	0	180
6644.24	0.025	40	0	0	180
6819.09	0.05	40	0	0	180
7003.39	0.075	40	0	0	180
7197.93	0.1	40	0	0	180
7621.34	0.15	40	0	0	180
8097.67	0.2	40	0	0	180
8637.52	0.25	40	0	0	180

Table 10: List of COEs for Altitude of Perigee = 80 km

a (km)	e	*i* (deg)	Ω (deg)	ω (deg)	M (deg)
6490.59	0.005	40	0	0	180
6506.94	0.0075	40	0	0	180
6523.37	0.01	40	0	0	180
6623.73	0.025	40	0	0	180
6798.04	0.05	40	0	0	180
6981.77	0.075	40	0	0	180
7175.71	0.1	40	0	0	180
7597.81	0.15	40	0	0	180
8072.67	0.2	40	0	0	180
8610.85	0.25	40	0	0	180

Using these COEs and the code produced the following plots for amount of time on orbit based on the initial altitude of apogee.

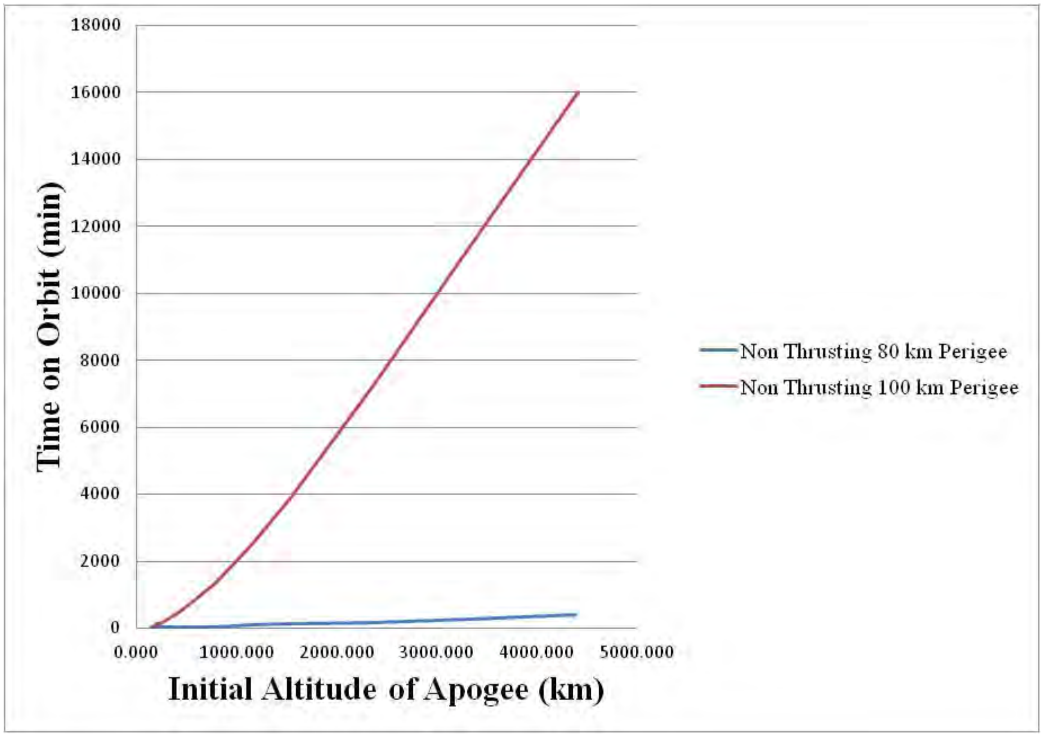


Figure 17: Amount of Time on Orbit for Non-Thrusting Satellite

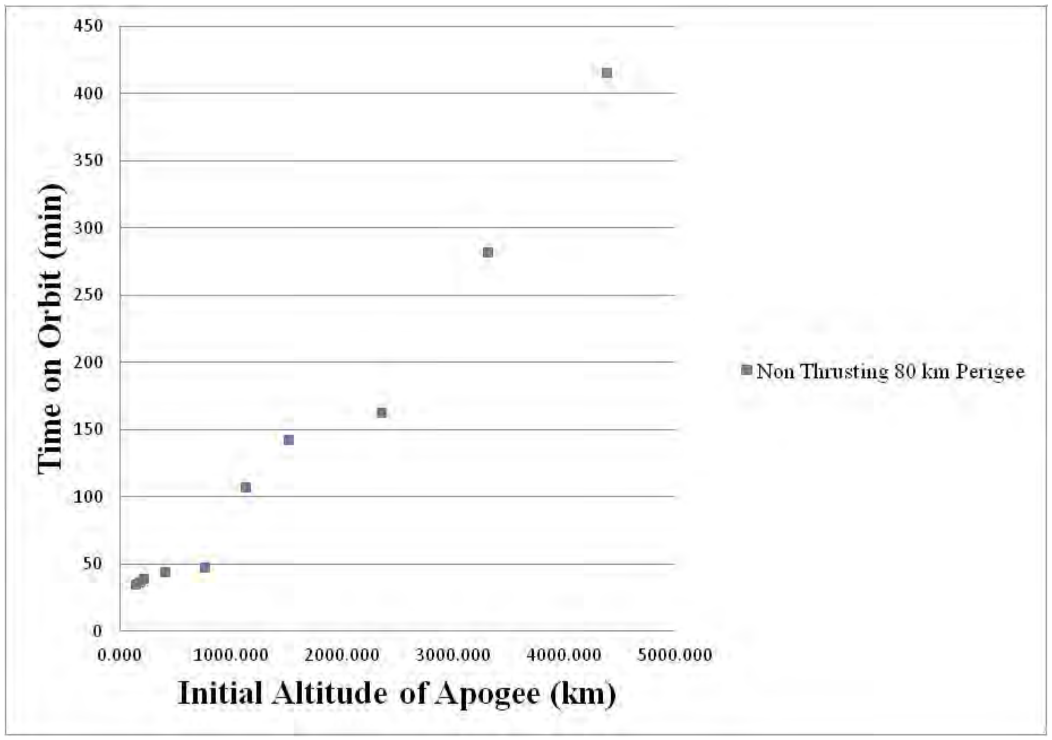


Figure 18: Amount of Time on Orbit for Non-Thrusting Satellite, 80 km Perigee

As Figure 17 and Figure 18 show smaller initial altitude of apogees have a lower amount of time on orbit due to the increased amount of time spent in the atmosphere and decreased time between perigee passes. An important feature of the graphs to consider is the two bends in the curves. If the reader looks at the 80 km perigee orbit in Figure 18 it can be seen the time on orbit follows an almost horizontal path until an altitude of apogee of ~1127 km. Once this altitude of apogee is reached the time on orbit quickly increases to roughly 150 minutes which corresponds to the second orbit's perigee crossing. At this point the satellite begins to follow another horizontal path until an altitude of apogee of ~2359 km is reached and the time on orbit begins to increase again. These horizontal paths are due to the satellite not having enough energy to overcome the first and second perigee crossings. Once the necessary energy is reached by increasing the initial altitude of apogee the satellite is able to make it through the atmosphere before drag causes a perigee collapse. These horizontal legs are not seen for the 100 km perigee orbit due to the higher initial perigee altitude resulting in reduced drag and an increased buffer before perigee collapse.

As a whole the larger initial altitude of apogees provide for longer durations in orbit, but this is at a cost of longer orbital periods. These longer orbital periods may or may not be suitable for ORS type missions. In the case of the 100 km perigee altitude an initial altitude of apogee of 4419 km the orbital period is more than 50% greater than the orbit with an initial altitude of apogee of 165 km. The next phase of the analysis will look at using a one Newton thruster to provide a boost to the orbit to correct for any loss in semi-major axis due to drag at low altitude perigees.

One Newton Thruster Booster

This section focuses on using a one Newton thruster to correct for the loss of altitude due to drag. The thruster would use a small constant thrust to increase the semi-major axis back to the semi-major axis value before drag caused a loss of semi-major axis at the perigee crossing. The first step is to find the amount of semi-major axis lost due to the drag during a perigee crossing. This is done by subtracting the semi-major axis value at a true anomaly of 45 degrees from the semi-major axis at 325 degrees. These points were chosen in order to incorporate as much of the altitude loss due to drag as possible while also providing a large enough true anomaly band for the initial burn after the first perigee crossing.

To simplify finding the true anomaly points at which to conduct the burns, a change of variable from time to eccentric anomaly was used. This eliminated the need to find the time at which the desired true anomaly point occurred. The change of variable was accomplished by using Equation 55 below.

$$\frac{dE}{dt} = \frac{1}{r}\sqrt{\frac{\mu}{a}} \tag{55}$$

By using this equation and the time rate of change for semi-major axis Equation 56 is produced.

$$\frac{da}{dE} = \frac{2a^3}{\mu}\left(a_r esinE + a_s\sqrt{1-e^2}\right)\sqrt{\frac{\mu}{a}} \tag{56}$$

If we assume a control angle, α, a burn arc can be produced for perigee, -α < E < α, and apogee, (π-α) < E< (π+α). Since apogee burns will be used for this analysis Equation 56 can be integrated over π – α to π + α to produce the following formula

$$\Delta a = \int_{\pi-\alpha}^{\pi+\alpha}\left(\frac{2a^3 a_s}{\mu}\sqrt{1-e^2}\right)dE = \left(\frac{4a^3 a_s}{\mu}\sqrt{1-e^2}\right)\alpha \tag{57}$$

Using the change in semi-major axis due to drag at perigee a control angle can be found and from that a burn arc. This method is used throughout the following analysis of perigee height maintenance.

The amount of time on orbit plots are repeated using the one Newton thruster and are shown below.

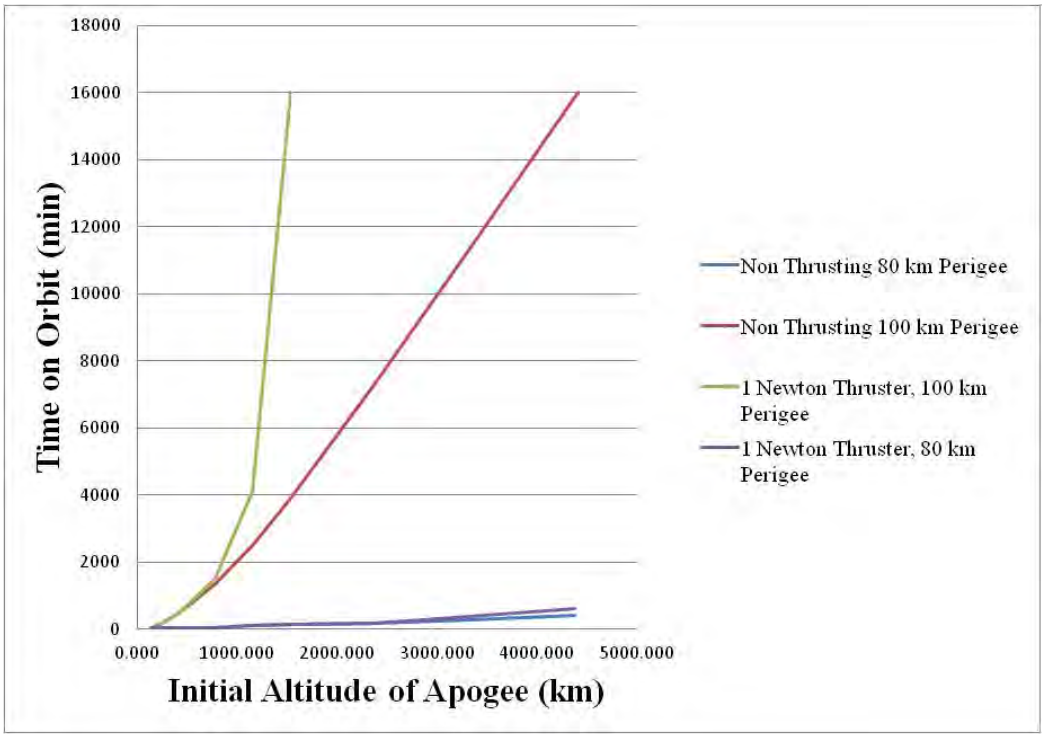


Figure 19: Amount of Time on Orbit for One Newton Thruster

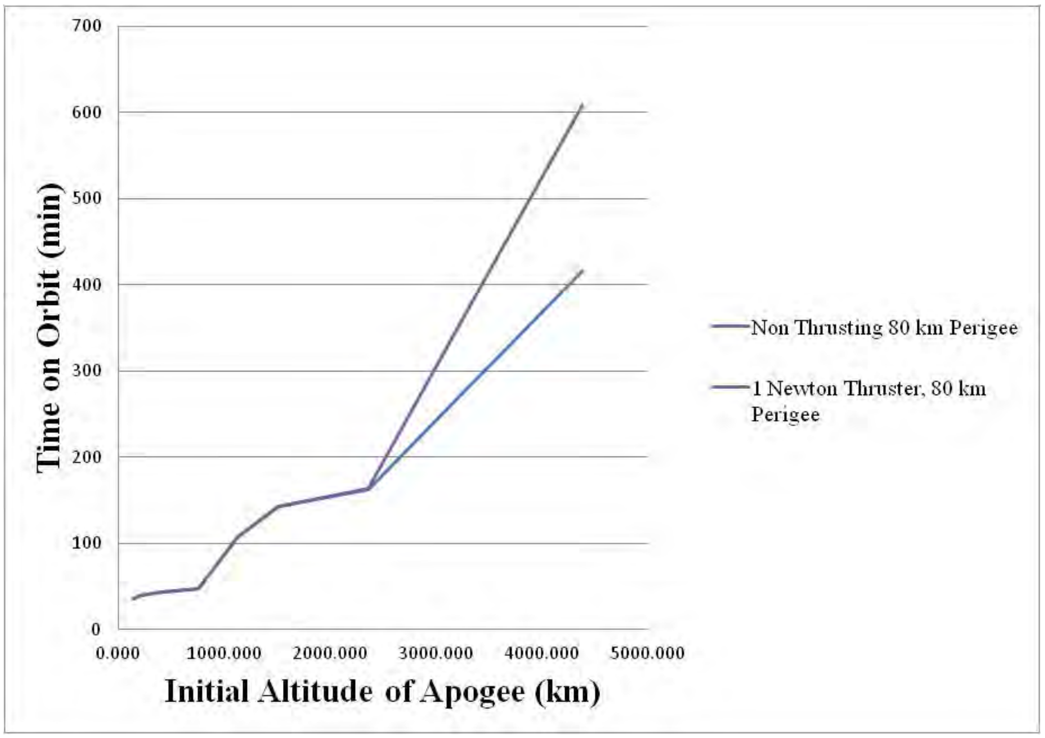


Figure 20: Amount of Time on Orbit for One Newton Thruster, 80 km Perigee

Based on the plots, each value showed an increase in amount of time on orbit. For the thrusting orbit with a 100 km perigee altitude the time on orbit reaches a peak of 16,000 minutes or 11.11 days at larger initial altitudes of apogee due to the code having a hard limit at this point. If the limit were increased, the time on orbit would also go up until the drag at perigee causes perigee collapse no matter how fast the satellite is going. In all other graphs the larger altitude of apogee data points will be removed if they go over 16,000 minutes in order to better show the trending. The simulations were not propagated out further due to the simulations with larger eccentricities taking upwards of 10 minutes of computing time to complete for the 60 second time step. This thesis is simply looking for the trends and the current time limit of 16000 minutes allows for analysis of these trends without unnecessary computation time.

For the orbits with a perigee of 80 km the satellites followed the same path as the non-thrusting orbits until an initial altitude of apogee of ~2359 km. At this point, the orbit had enough speed to overcome the second perigee crossing and had a steady growth as the initial altitude of apogee increases. In order for the reader to better understand what is occurring to the orbit as time progresses the altitude of perigee and apogee were plotted for the 80 km perigee satellite with its largest initial apogee height of 4385 km in Figure 21 and Figure 22 on the next page.

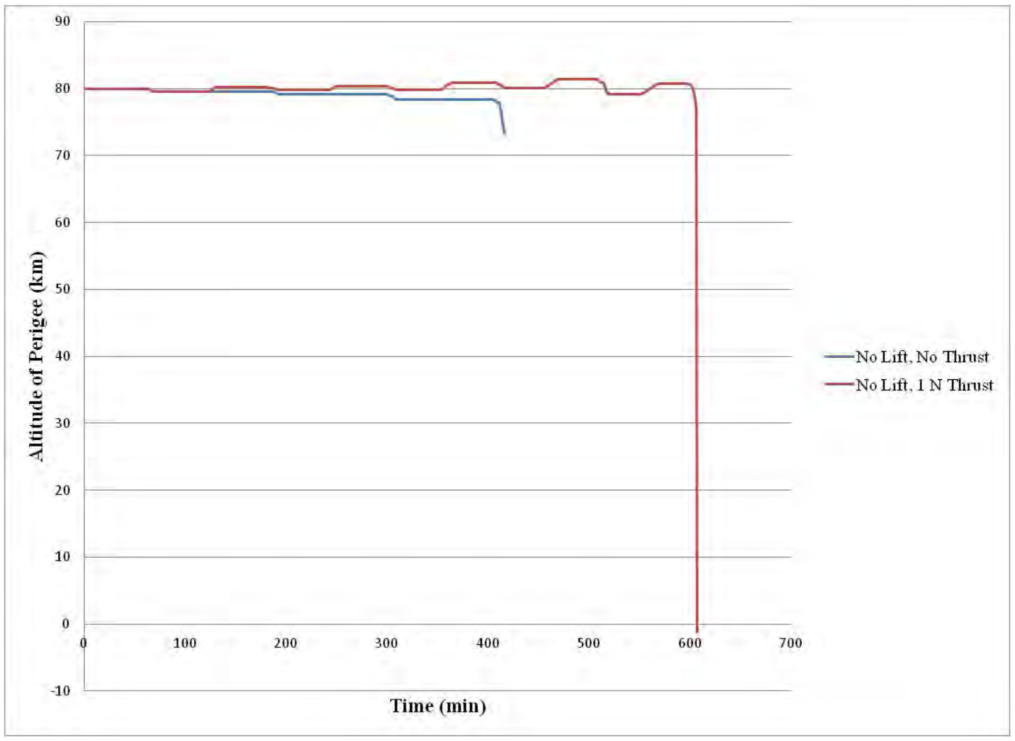


Figure 21: Altitude of Perigee vs. Time, 80 km Perigee, 4385 km Apogee

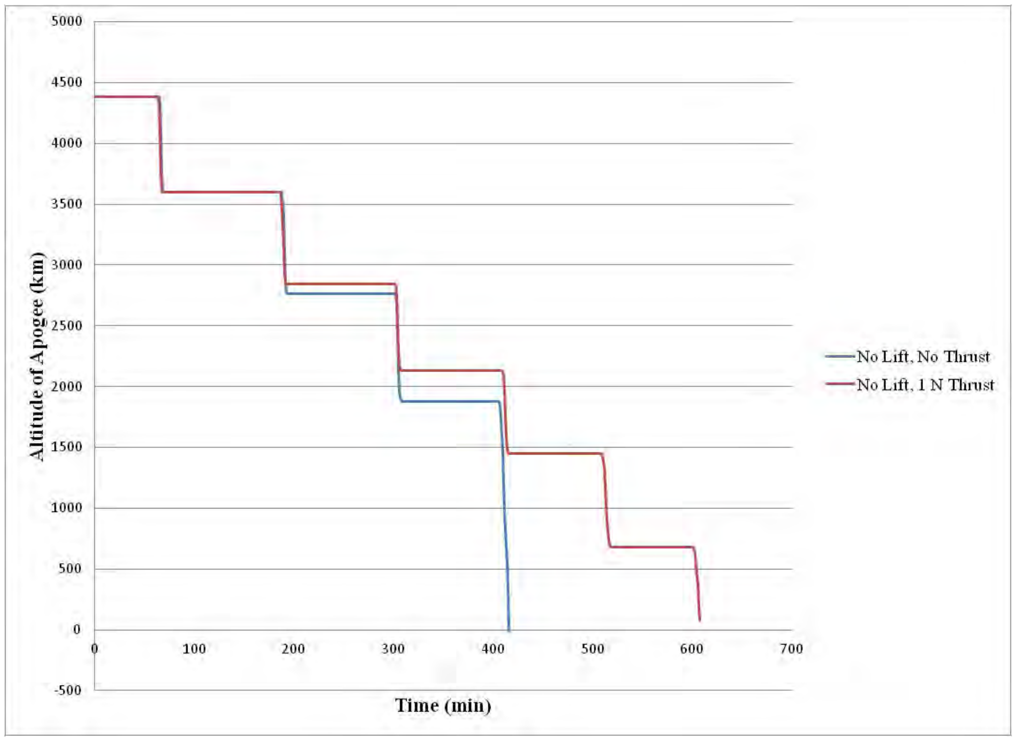


Figure 22: Altitude of Apogee vs. Time, 80 km Perigee, 4385 km Apogee

As the reader can see the perigee height stays relatively the same with a slightly negative slope as time progresses for the non-thrusting satellite. For the thrusting satellite the perigee slightly increases due to the thrust at apogee but as the apogee decreases the drag at perigee cancels this increase from the thruster out. As a result the perigee height stays slightly over 80 km until perigee collapse occurs. For the apogee height both the satellites have a negative trend as the drag at each perigee crossing slowly circularizes the orbit and brings apogee down. Each horizontal line is the time between perigee crossings and the vertical lines is the drop off in apogee caused by drag at perigee. The orbits stay the same up until the second perigee crossing as the satellites start at apogee and do not fire until the next apogee. This causes the thrusting satellite to have less apogee decrease for the second perigee crossing and ultimately allows the thrusting satellite to have two additional perigee crossings before perigee collapse when compared with the non-thrusting satellite.

For the 100 km perigee orbits the satellites followed the same path as the non-thrusting satellites until an initial altitude of apogee of 432 km. After this they had a steady growth, which quickly transitioned into an exponential growth for time on orbit. Again, for the readers benefit the altitude of perigee and apogee are plotted for the 100 km perigee orbit. The initial apogee height of 432 km chosen since this is where the transition occurs between the thrusting and non-thrusting satellites. These plots are shown in Figure 23 and Figure 24 on the next page.

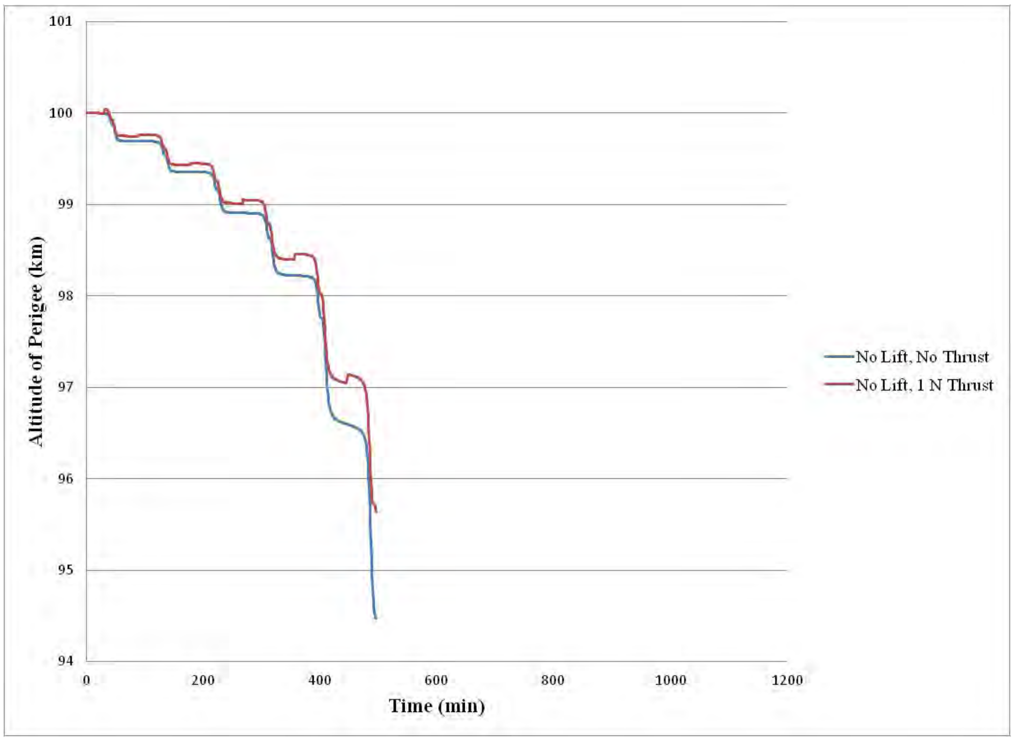


Figure 23: Altitude of Perigee vs. Time, 100 km Perigee, 432 km Apogee

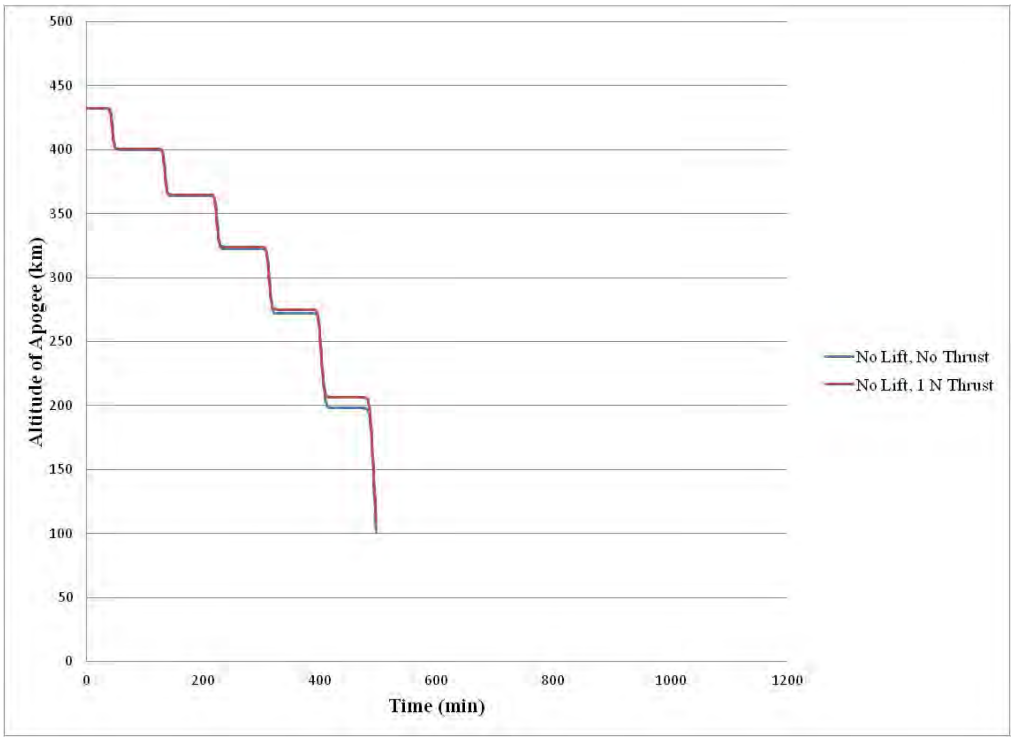


Figure 24: Altitude of Apogee vs. Time, 100 km Perigee, 432 km Apogee

These plots generate the same trend as seen before with the 80 km perigee height. The thrusting orbit is just beginning to improve the apogee height as perigee collapse occurs. The thrusting only provides a slight increase in perigee altitude when compared with the decrease due to drag.

One important note to take into account is the slight increase in perigee height on right before the first perigee crossing. This is due to the thruster burning for one iteration because of the author's poor choice of using 0 to initialize the beginning semi-major axis for the thruster firing loop. This choice of 0 caused the thruster firing loop to have a large control angle. Once the thruster fired and the beginning semi-major axis was changed to a reasonable value of approximately 6500 km, the thruster firing loop behaved correctly. This error was not noticed until corrections were made for the final copy and the perigee height graphs were included into the thesis. Since the reason for the slight increase was understood and it did not affect the trending; the decision to use the slightly perturbed data was made.

It is also important to note the disparity in time on orbit attained by the different initial perigee altitudes for the simulations. For an initial altitude of apogee of 1540 km the 100 km perigee orbit reached the time limit of 16,000 minutes compared to 3,900 minutes for the non-thrusting 100 km perigee orbit and 143 minutes for the 80 km perigee orbit at the same initial altitude of apogee. At these points the thrusting 100 km orbit burns all 200 kg of the fuel while the thrusting 80 km orbit re-enters the atmosphere after only burning 3 kg of fuel. Since the non-thrusting 100 km orbit was able to achieve 3,900 minutes with no fuel use and the thrusting 80 km orbit only completed achieved 143 minutes with fuel it seems much more useful to have a few kilometers of perigee height than a small constant thrust engine.

This small increase in perigee provided a 27-fold increase for time on orbit for the non-thrusting satellites. Using the same comparison of time on orbit for the non-thrusting and thrusting 100 km orbits shows a four-fold increase for time on orbit with the trade being the 200 kg of fuel used. This is still a significant increase in time on orbit. Having observed the usefulness of a constant thrust engine in enabling low altitude orbits, the next step is to observe the added benefit of lift a satellite could incorporate utilizing a space plane design.

Addition of Lift

The specifications chosen for the space plane design as used earlier in the MatLab/STK comparison are shown in Table 11 below.

Table 11: Space Plane Design Specifications

Parameter	Value
Area (m^2)	2.67
Coefficient of Drag	1.0
Wet Mass (kg)	2000
Flight Path Angle/Angle of Attack (degrees)	2

In addition to the specifications above the coefficient of lift was run at 0.5, 1.0, and 1.5. The space plane will also use the same thruster as before and utilize the same algorithm for determining the burn duration.

Figure 25 helps the reader visualize the direction of the forces the space plane will experience while traveling through the atmosphere.

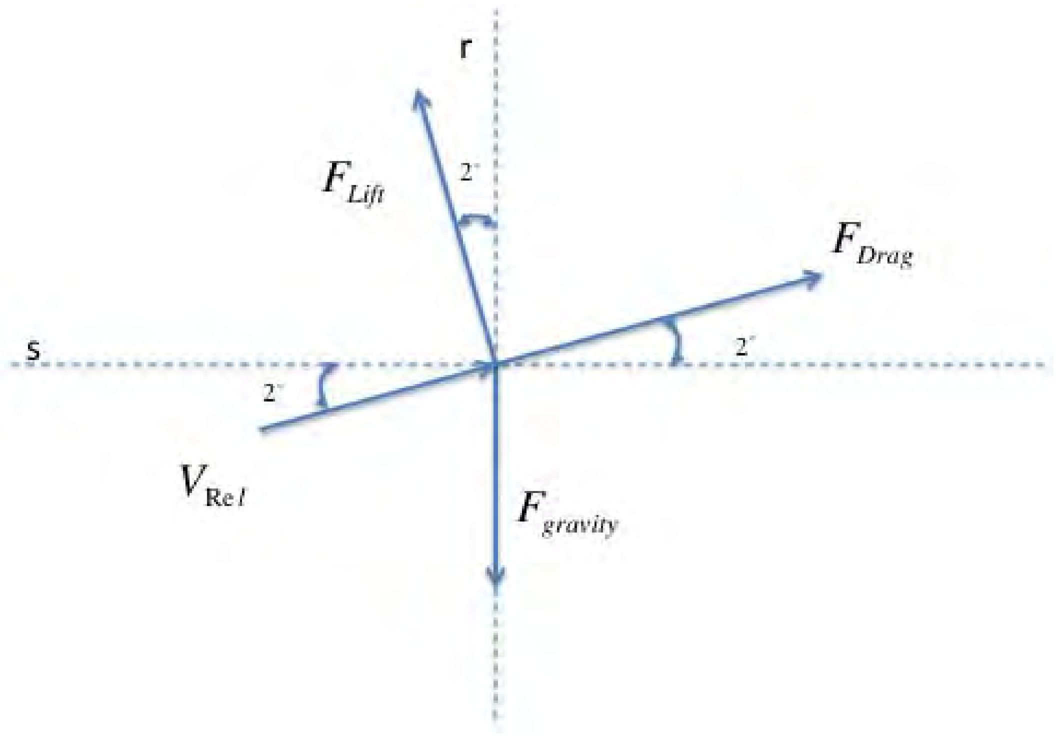


Figure 25: Direction of Forces

In order to simplify the problem, the flight path angle/angle of attack for the space plane is set at two degrees and does not change throughout the orbit propagation. This constant flight path angle ensures the L/D ratio will not change and allows for a simpler code creation in order to understand the trends. In reality these values could change depending on perturbations and the stability of the space plane design. An attitude control system would be needed in order to ensure these values did not change.

Based on Figure 25, Figure 12, and Figure 13 above the lift force will provide a positive change in semi-major axis due to the tangential force during the entire perigee crossing. For the radial component of the lift force a positive change in semi-major axis will be experienced while the satellite is heading away from perigee and a negative change in semi-major axis while the satellite is heading towards perigee. For drag, the radial component will provide the same direction (positive or negative) of changes as the lift force during the same true anomaly bands, however the change in semi-major axis due to the tangential force will always be negative. Simulations were run for lift to drag ratios of 0.5, 1, and 1.5 using the same COEs used in the previous sections and the space plane specifications listed above. The amount of time on orbit plots are shown below for each lift to drag ratio along with the thruster and non-thrusting simulations.

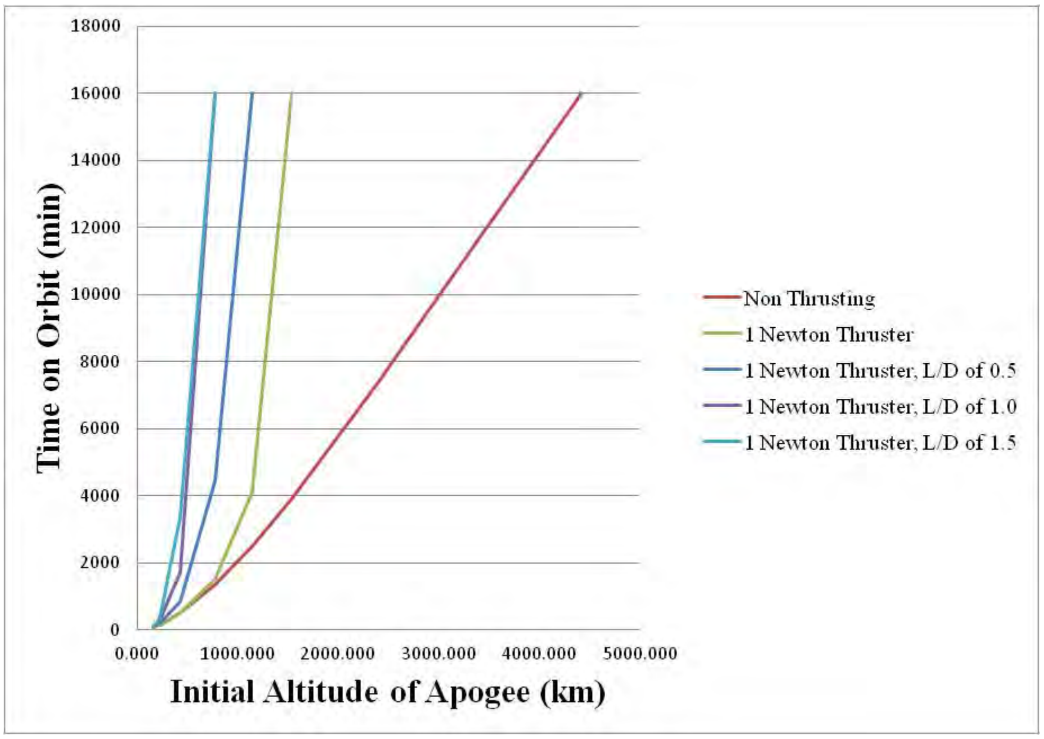


Figure 26: Amount of Time on Orbit for One Newton Thrust, Various L/D's, and 100 km Perigee

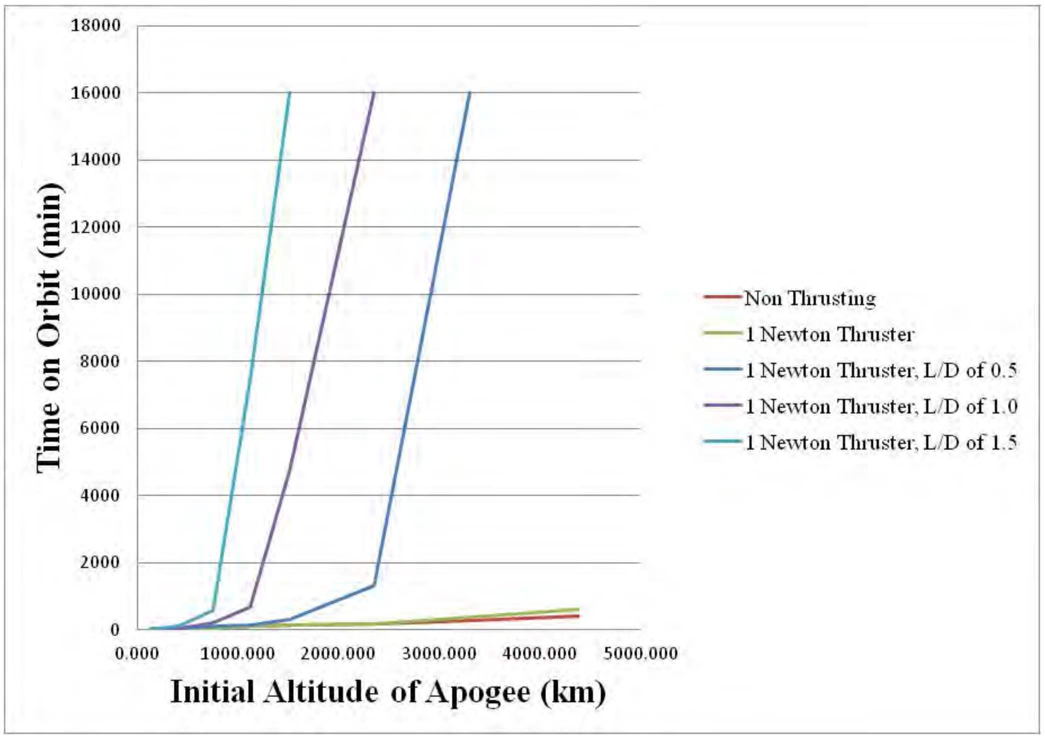


Figure 27: Amount of Time on Orbit for One Newton Thrust, Various L/D's, and 80 km Perigee

Based on the plots a lift force greatly increases the amount of time and number of orbits a satellite can achieve with unsurprisingly the larger L/D ratio producing the best results. Both the 80 km and 100 km simulations show a vast improvement in time on orbit. These improvements are shown as multiples of the non-thrusting time on orbit values in Table 12 and Table 13 on the next pages. The N/A's are put in place because of unreliable data due to the hard time stop of 16,000 minutes implemented in the MatLab code. The thrusting orbits have a value of 1.00 for many of the simulations in both tables due to the satellite not having enough speed to overcome the perigee collapse. Since the thruster only provides a small increase in speed and the time step used is 60 seconds they appear to have the same collapse time. In reality, the thrusting satellite would stay on orbit for a fraction of a minute longer. It should also be noted that the addition of lift also causes the horizontal sections where perigee collapse occurs for many of the initial altitude of apogee points to occur sooner. For example the 80 km perigee orbit sees the second perigee crossing occur at 760 km for the L/D of 0.5, somewhere between 760 and 411 km for the L/D of 1.0, and 411 km for the L/D of 1.5. Another important note is the 100 km perigee orbit for an initial altitude of 198 km and an L/D ratio of 0.5 showing a decrease in time on orbit. This decrease occurs for the same reason. If more initial altitude of apogee points were used the graph would show the satellite with an L/D ratio of 0.5 has its second perigee crossing point at lower initial altitude of apogee values.

Table 12: 100 km Perigee Time on Orbit Improvements

Initial Altitude of Apogee (km)	Non-Thrusting (min)	Thrusting	L/D of 0.5	L/D of 1.0	L/D of 1.5
165.107	48	1.00	2.15	2.40	2.44
197.906	130	1.00	0.98	1.09	1.60
230.871	136	1.00	1.61	2.23	2.80
432.212	496	1.00	1.70	3.44	6.80
781.909	1340	1.13	3.31	N/A	N/A
1150.509	2497	1.65	N/A	N/A	N/A
1539.586	3900	N/A	N/A	N/A	N/A
2386.401	7334	N/A	N/A	N/A	N/A
3339.069	11407	N/A	N/A	N/A	N/A
4418.758	16000	N/A	N/A	N/A	N/A

Table 13: 80 km Perigee Time on Orbit Improvements

Initial Altitude of Apogee (km)	Non-Thrusting (min)	Thrusting	L/D of 0.5	L/D of 1.0	L/D of 1.5
144.906	35	1.00	1.00	1.03	1.06
177.604	37	1.00	1.03	1.05	1.00
210.467	39	1.00	1.03	1.00	1.05
411.187	44	1.00	1.02	1.07	2.55
759.804	48	1.00	2.58	4.38	11.90
1127.265	107	1.01	1.33	6.39	69.71
1515.142	143	1.00	2.24	33.19	N/A
2359.342	163	1.01	8.20	N/A	N/A
3309.069	282	1.32	N/A	N/A	N/A
4385.425	416	1.46	N/A	N/A	N/A

The same plots for altitude of perigee and apogee were generated with the addition of lift included in the particular initial apogee altitudes chosen. These graphs are shown below and on the next few pages.

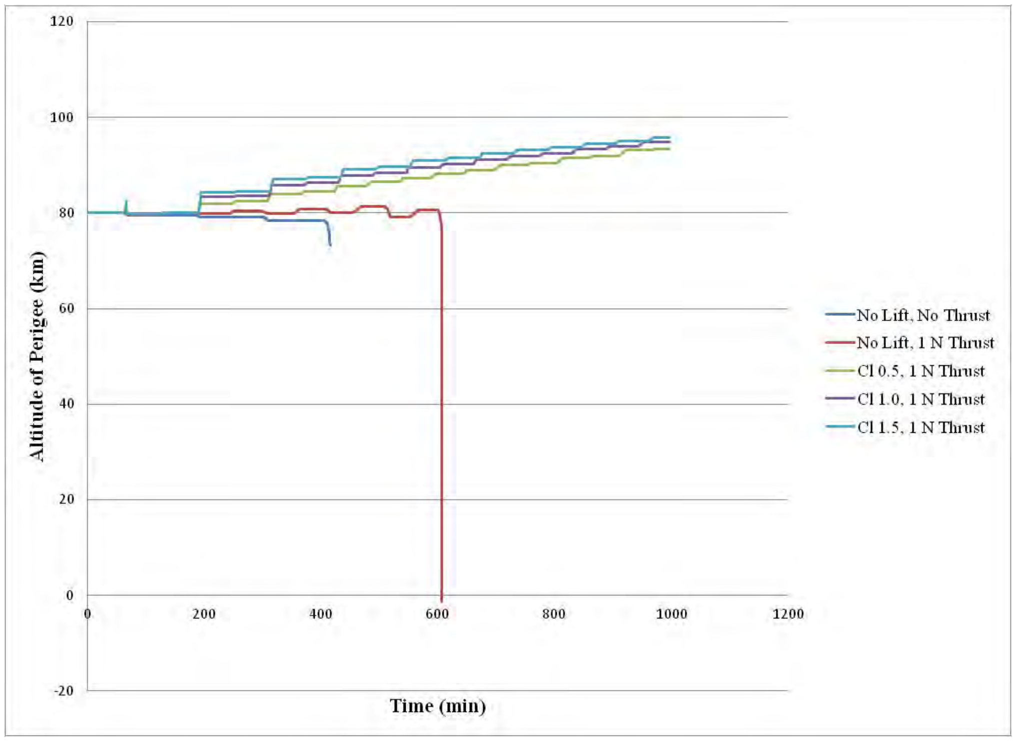


Figure 28: Altitude of Perigee vs. Time, 80 km Perigee, 4385 km Apogee

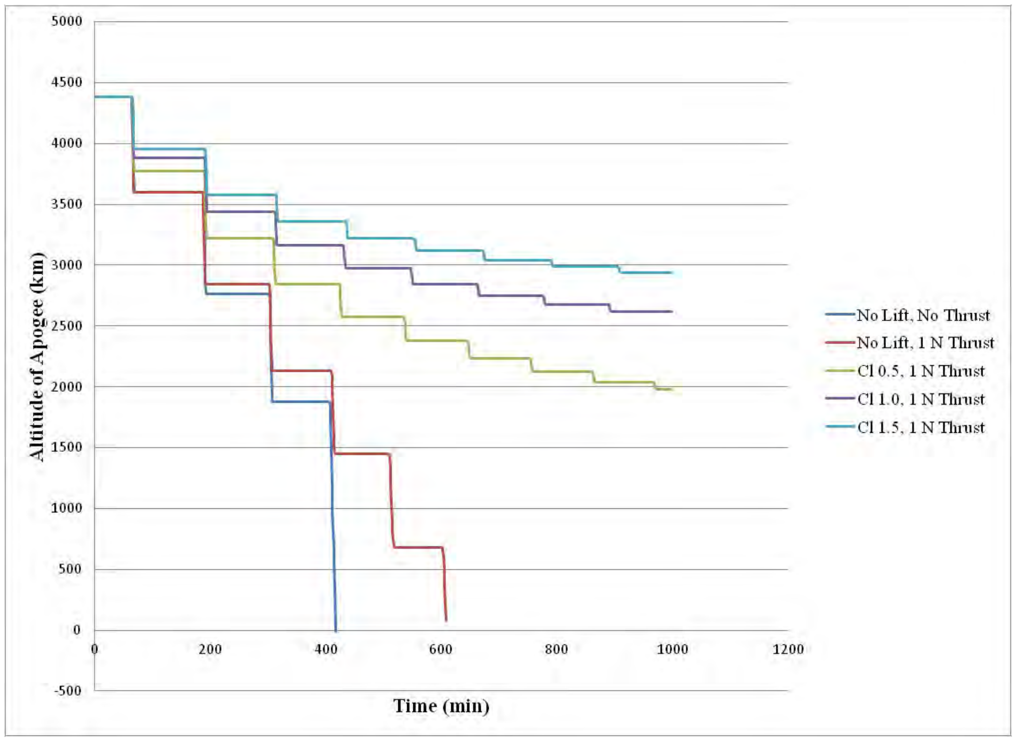


Figure 29: Altitude of Apogee vs. Time, 80 km Perigee, 4385 km Apogee

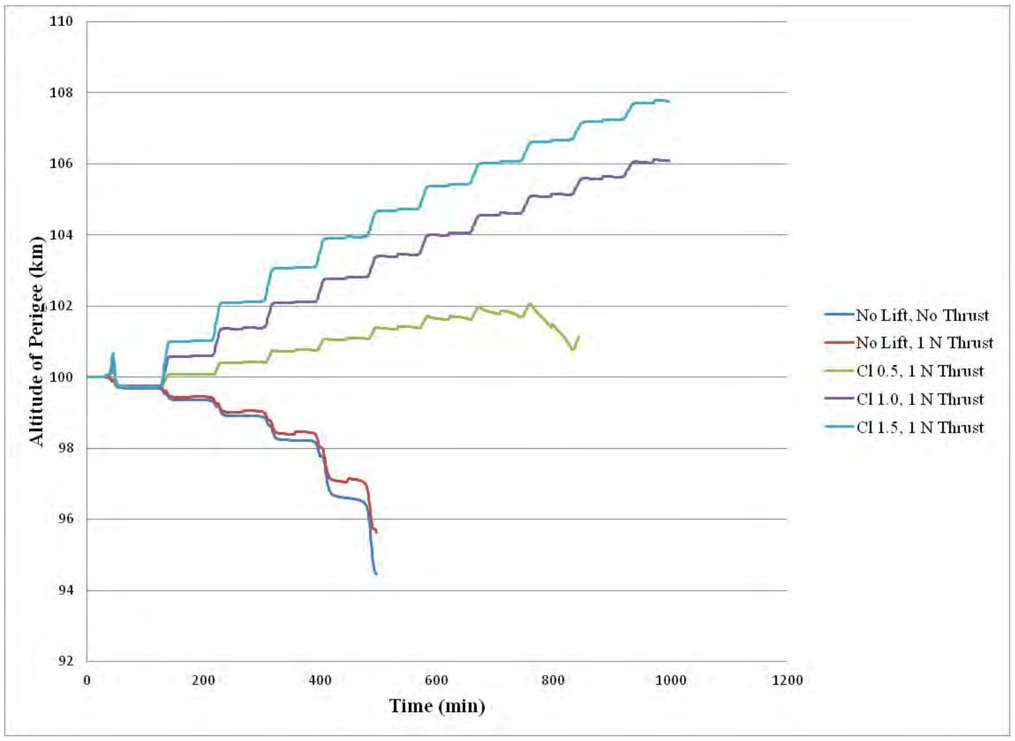


Figure 30: Altitude of Perigee vs. Time, 100 km Perigee, 432 km Apogee

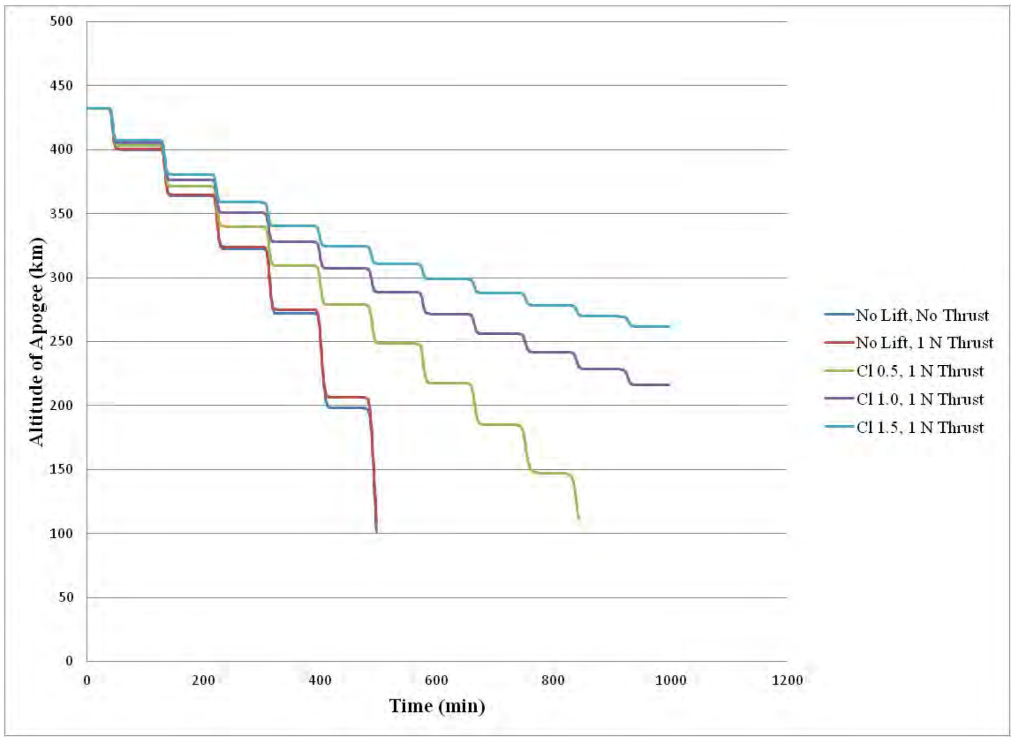


Figure 31: Altitude of Apogee vs. Time, 100 km Perigee, 432 km Apogee

The same increase in perigee height before the first perigee crossing is noticed in the lift simulations as well due to the thruster using the same thruster firing loop. As lift is included into the simulations, an increase in perigee height is noticed. This increase in perigee height is caused by the decrease of eccentricity rather than an increase in semi-major axis. In fact if the reader finds semi-major axis using the perigee and apogee height he/she will notice an overall decrease in semi-major axis. As the perigee height increases the loss in apogee height gets smaller and smaller as now that the perigee height is higher the affect of drag is reduced due to the lower air density. The simulation with an initial perigee height of 100 km and a lift coefficient of 0.5 shows the orbit on the verge of perigee collapse. Notice the sharp decrease in perigee height once the apogee height has dropped below 150 km. The thruster firings are only slightly noticeable in the perigee height graphs as the slight humps in the middle of horizontal legs of the graphs.

Addition of Lift with No Thruster Support

After observing the large increase in time on orbit due to the incorporation of lift with a satellite using a one Newton thruster, further simulations were conducted with no thruster support for the 100 km perigee orbit utilizing the same L/D and space plane profiles. The same plot profiles were used and are shown below. Each plot shows the non-thrusting orbit, the thrusting orbit, and the specified L/D ratio orbits with and without a thruster.

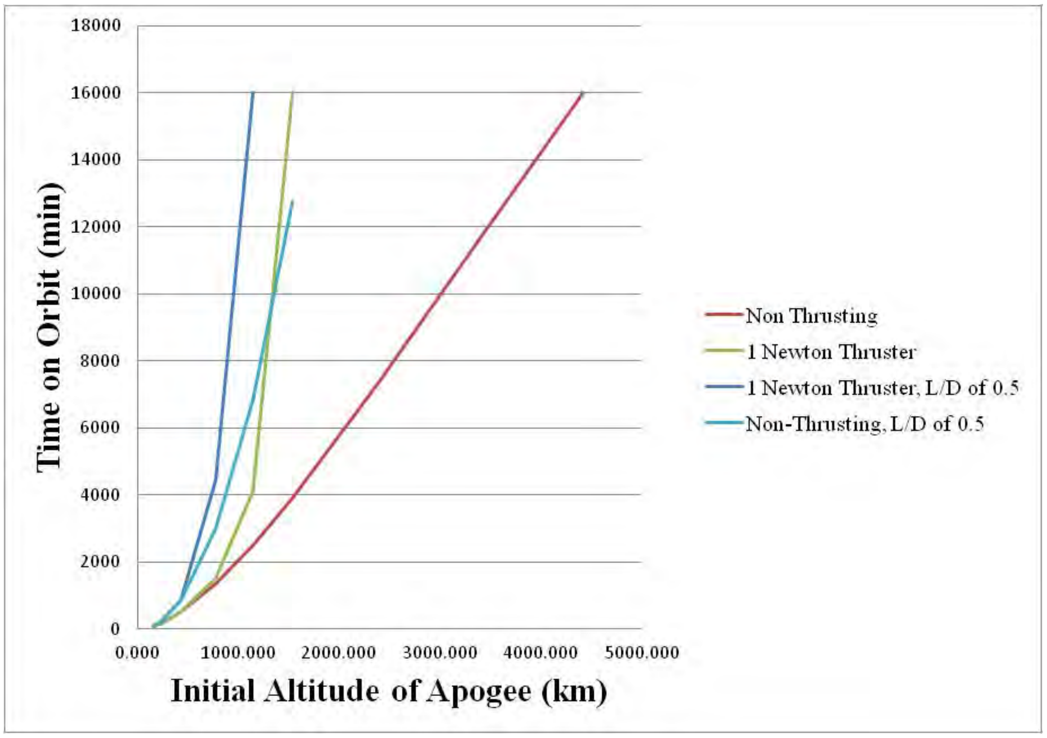


Figure 32: Amount of Time on Orbit for Non-Thrusting Satellite and L/D of 0.5

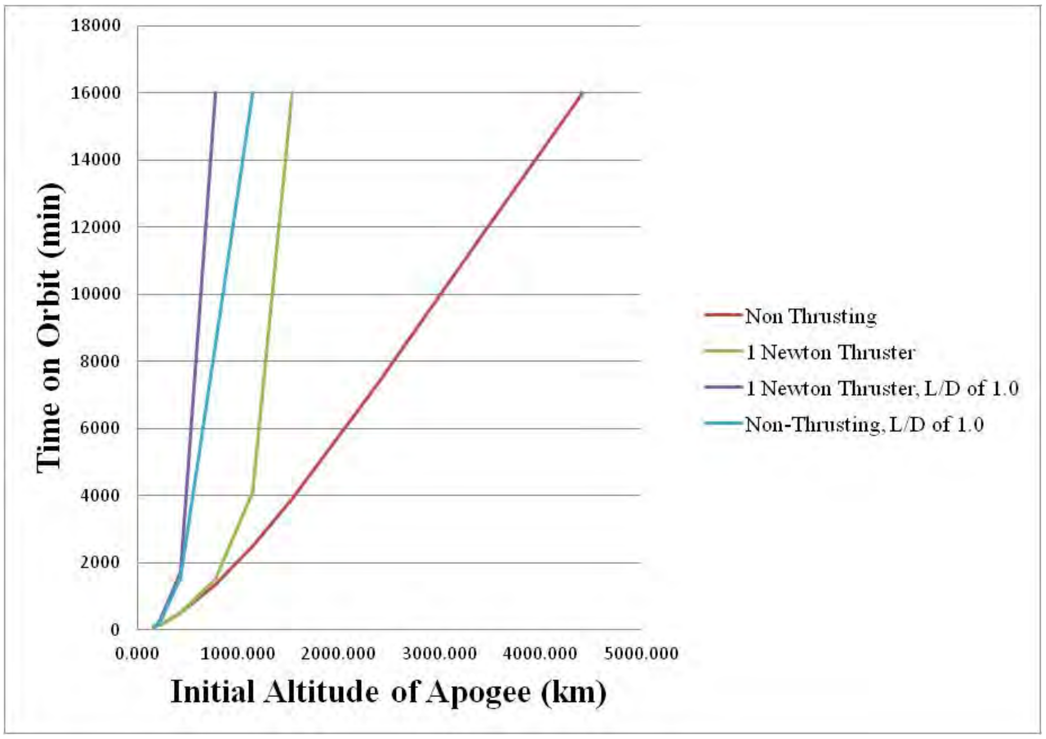


Figure 33: Amount of Time on Orbit for Non-Thrusting Satellite and L/D of 1.0

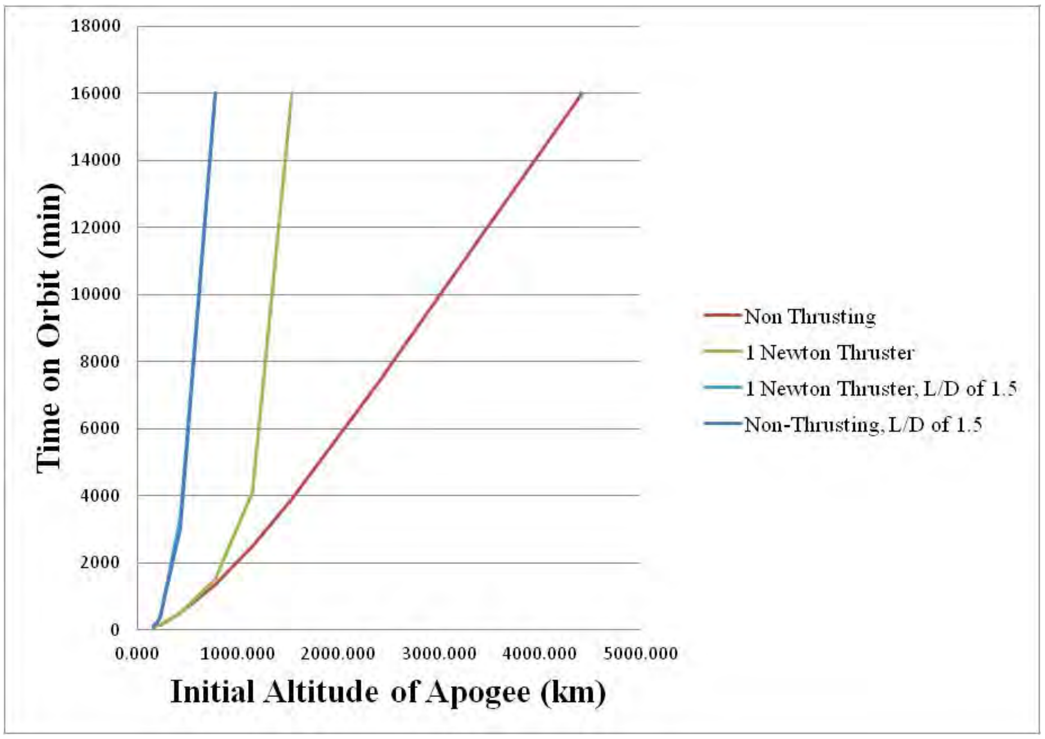


Figure 34: Amount of Time on Orbit for Non-Thrusting Satellite and L/D of 1.5

As the charts show the simulations without the thruster still provide significant increases for time on orbit. The table showing the percentage comparisons for the 100 km perigee orbit is again shown below with the addition of the simulations without the thruster percentages shown in parenthesis.

Table 14: 100 km Perigee Time on Orbit Improvements

Initial Altitude of Apogee (km)	Non-Thrusting (min)	Thrusting	L/D of 0.5	L/D of 1.0	L/D of 1.5
165.107	48	1.00	2.15 (2.08)	2.40 (2.38)	2.44 (2.44)
197.906	130	1.00	0.98 (0.98)	1.09 (1.07)	1.60 (1.60)
230.871	136	1.00	1.61 (1.60)	2.23 (1.73)	2.80 (2.82)
432.212	496	1.00	1.70 (1.69)	3.44 (3.15)	6.80 (6.12)
781.909	1340	1.13	3.31 (2.25)	N/A	N/A
1150.509	2497	1.65	N/A	N/A	N/A
1539.586	3900	N/A	N/A	N/A	N/A
2386.401	7334	N/A	N/A	N/A	N/A
3339.069	11,407	N/A	N/A	N/A	N/A
4418.758	16,000	N/A	N/A	N/A	N/A

The same improvement in time on orbit trends can be seen for the simulations without the thruster support. The multiples are simply smaller due to the loss of the additional energy the thruster could provide. For smaller initial altitude of apogees the thruster does not provide a large increase when compared with the non-thruster simulations. However, the larger initial altitudes of apogee do show significant improvement from the thruster. The L/D ratio of 1.5 at an initial altitude of apogee of 231 km shows the non-thrusting satellite as having a longer time on orbit; this is due to the second perigee crossing occurring sooner for the thrusting simulation vs. the non-thrusting simulation.

Equivalent Fuel Usage

The next step in analyzing the lift forces was to determine how much fuel would be saved had the thruster been used to compensate rather than using lift. By using the average force at each time step a total impulse could be found by multiplying by the time step, 60 seconds. This total impulse could then be divided by the specific impulse (200) and acceleration due to gravity to find the equivalent mass of propellant. It is important to keep in mind the average force at a particular time step is a combination of the R, S, and W component forces. The propellant masses over the course of the entire orbit could then be summed up to find the total equivalent propellant that would need to be expelled by the thruster to match the force generated by lift. The total equivalent propellant mass and the total mass actually used by the thruster for each simulation are tabulated in Table 15 and Table 16 below and on the next page. As the tables show the addition of force provides a substantial savings in fuel and at lower initial altitudes the lift is doing most of the work in keeping the satellite on orbit.

Table 15: 80 km Perigee Equivalent Fuel Usage

Initial Altitude of Apogee (km)	L/D of 0.5		L/D of 1.0		L/D of 1.5	
	Thruster Propellant	Equivalent Propellant	Thruster Propellant	Equivalent Propellant	Thruster Propellant	Equivalent Propellant
144.906	0.000	15.707	0.000	34.998	0.000	52.428
177.604	1.162	25.179	1.193	51.166	1.022	38.770
210.467	1.223	35.477	1.010	41.214	1.007	73.796
411.187	1.012	62.583	1.012	90.444	2.333	167.348
759.804	2.610	120.811	5.258	201.289	15.767	286.763
1127.265	2.852	157.141	18.876	289.883	199.453	421.473
1515.142	8.545	196.154	143.416	381.171	197.712	440.828
2359.342	39.618	286.955	199.552	390.768	197.241	461.520
3309.069	199.218	275.918	197.320	385.332	197.347	467.996
4385.425	199.824	253.522	198.370	372.704	197.663	458.831

Table 16: 100 km Perigee Equivalent Fuel Usage

Initial Altitude of Apogee (km)	L/D of 0.5		L/D of 1.0		L/D of 1.5	
	Thruster Propellant	Equivalent Propellant	Thruster Propellant	Equivalent Propellant	Thruster Propellant	Equivalent Propellant
165.107	2.214	19.734	2.317	26.695	2.335	32.235
197.906	2.564	14.979	2.536	27.870	5.122	42.623
230.871	5.430	31.708	8.168	61.176	10.612	61.376
432.212	24.350	48.604	51.130	97.346	102.042	138.661
781.909	134.185	94.321	198.507	170.264	199.886	185.428
1150.509	199.221	98.814	198.314	146.846	199.402	181.730
1539.586	199.404	85.605	197.873	138.581	198.793	176.814
2386.401	198.606	71.603	199.562	123.380	197.011	165.164
3339.069	196.634	61.846	197.327	110.497	197.873	150.567
4418.758	197.274	53.363	197.812	97.873	198.250	136.444

Contact Windows

Determining whether the additional speed through the atmosphere limited the contact windows of the satellite was also deemed necessary to determine. To accomplish this study STK was used. The choice to use STK vs. the MatLab code used in this thesis was simply because the tools necessary to analyze contact windows is already implemented in STK.

Satellites with the same COEs used throughout this analysis of this paper were generated in STK and a target for the satellites to view was generated at Dayton, OH (39°45'32" N / 84°11'30" W). The average contact time was then used to determine if contact windows would be affected. The length of the contact window for each initial eccentricity is tabulated in Table 17 on the next page. It should be noted the orbital periods are not the same for each case as the initial altitude of apogee is changed; higher apogees will therefore have a longer orbital period.

Table 17: Contact Window Average Length

Initial Altitude of Apogee (km)	100 km Perigee (min)	Initial Altitude of Apogee (km)	80 km Perigee (min)
165.107	5.18	144.906	4.85
197.906	5.60	177.604	5.03
230.871	5.88	210.467	5.55
432.212	8.10	411.187	7.22
781.909	10.07	759.804	9.90
1150.509	12.60	1127.265	12.42
1539.586	14.90	1515.142	14.88
2386.401	20.75	2359.342	19.98
3339.069	25.82	3309.069	25.87
4418.758	33.28	4385.425	32.38

Heating

Aerodynamic heating is one of the most important issues to address in ensuring this approach of dipping into atmosphere is feasible. Although this thesis will not do an in depth analysis of the heating loads and the effects on the vehicle; trending will be studied using a non-dimensional approach presented by Dr. Hicks. A complete breakdown of his method can be found in his book, Aerodynamic Re-entry. The final equations for a skip re-entry are detailed below.

$$\Delta Q = \overline{Q_f}\left(\frac{1}{2}m\,{}^{R}V_0^2\right) \tag{58}$$

$$\overline{Q_f} = -\left(\frac{c_f A}{C_D S}\right) T_e \left\{ exp\left[\frac{4\gamma_e}{\left(\frac{C_L}{C_D}\right)}\right] - 1\right\} \tag{59}$$

$$T_e = \frac{1}{2}\left(\frac{{}^{R}V_0^2}{g_0 r_0}\right) \tag{60}$$

where: c_f = *the average skin coefficient*

A = the total vehicle surface area
C_D = the coefficient of drag
S = the "wetted" area of the vehicle
γ_e = flight path angle at entry
C_L = the coefficient of lift
T_e = the initial non-dimensional re-entry kinetic energy
g_0 = the force of gravity at sea level
r_0 = the radius of the Earth
${}^{R}V_0$ = the relative velocity of the vehicle at re-entry
m = the mass of the vehicle

This particular analysis will look at five different entry angles; 2,3,4,5, and 6 degrees, which are analogous to the flight path angle used earlier in this thesis. In order for the satellite to survive the repeated atmospheric re-entries from the low altitude perigee orbits the angle must be small. The reader can also estimate the effects of a larger re-entry angle by increasing the final heating loads discussed below by the percentage increase in the exponential of the new re-entry angle condition compared with the exponential of the old re-entry angle condition.

For simplicity the average skin friction coefficient is divided out, as this particular coefficient is difficult to model however typical values range from 0-2. The total surface area is assumed to be double that of the "wetted" area also for simplicity. Figure 35 shows $\Delta Q/c_f$ for the various L/D ratios versus the re-entry velocity. The mass of the vehicle is held constant at 2000 kg.

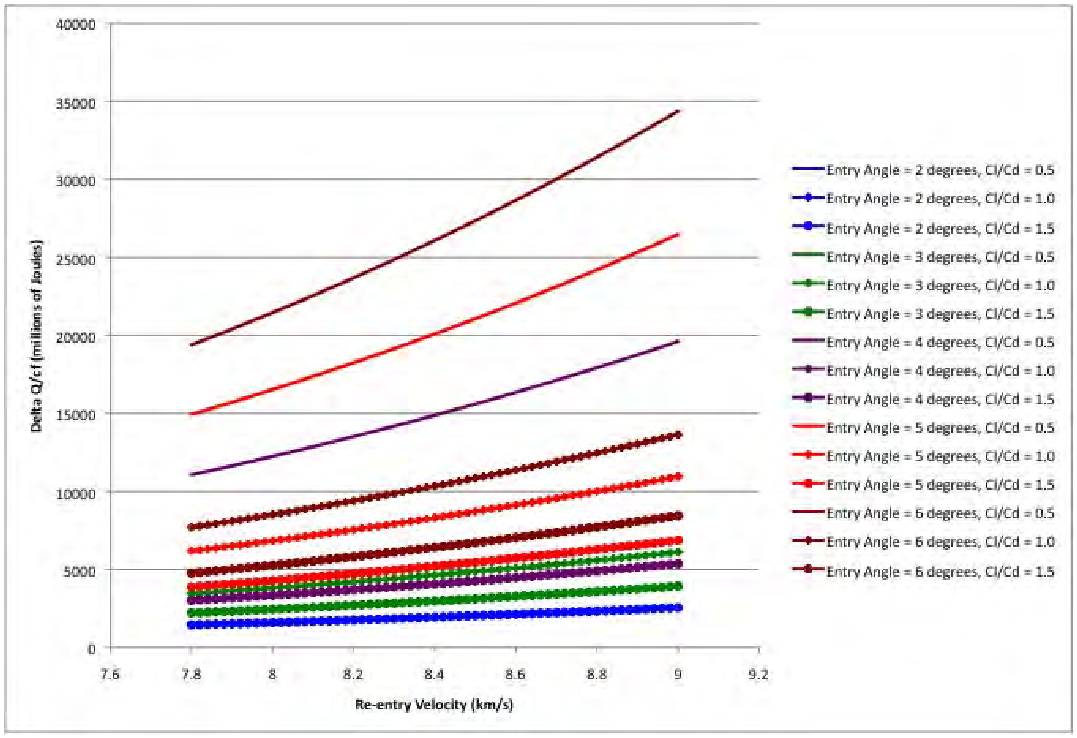


Figure 35: Different L/D Heating Loads Based on Re-entry Velocity

Figure 36, Figure 37, and Figure 38 provide a closer look at the lower heating lines and remove some of the lines to show the locations of overlapping lines not visible on the graph above.

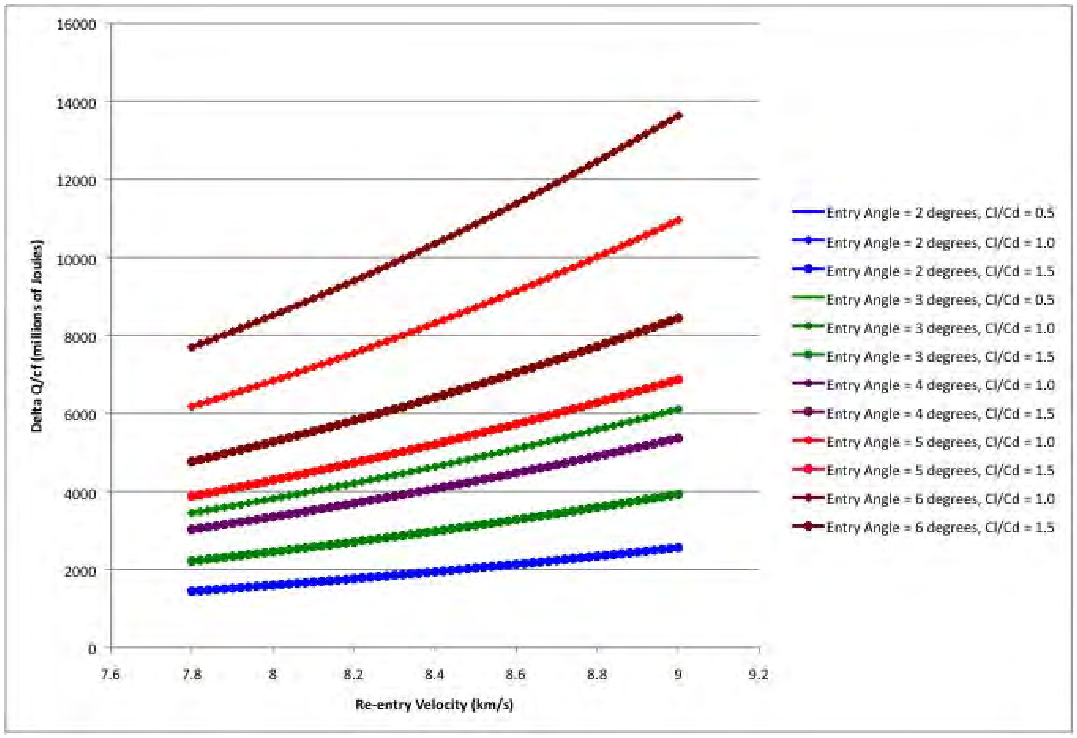


Figure 36: Closer Look at Lower Heating Loads (1)

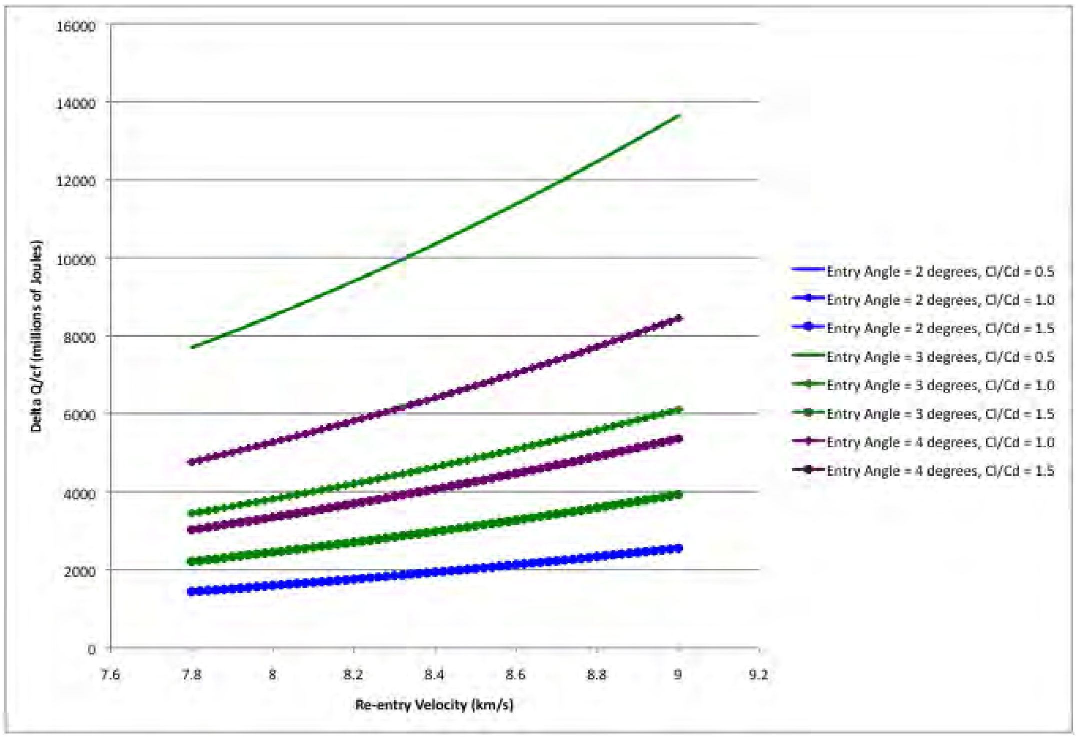


Figure 37: Closer Look at Lower Heating Loads (2)

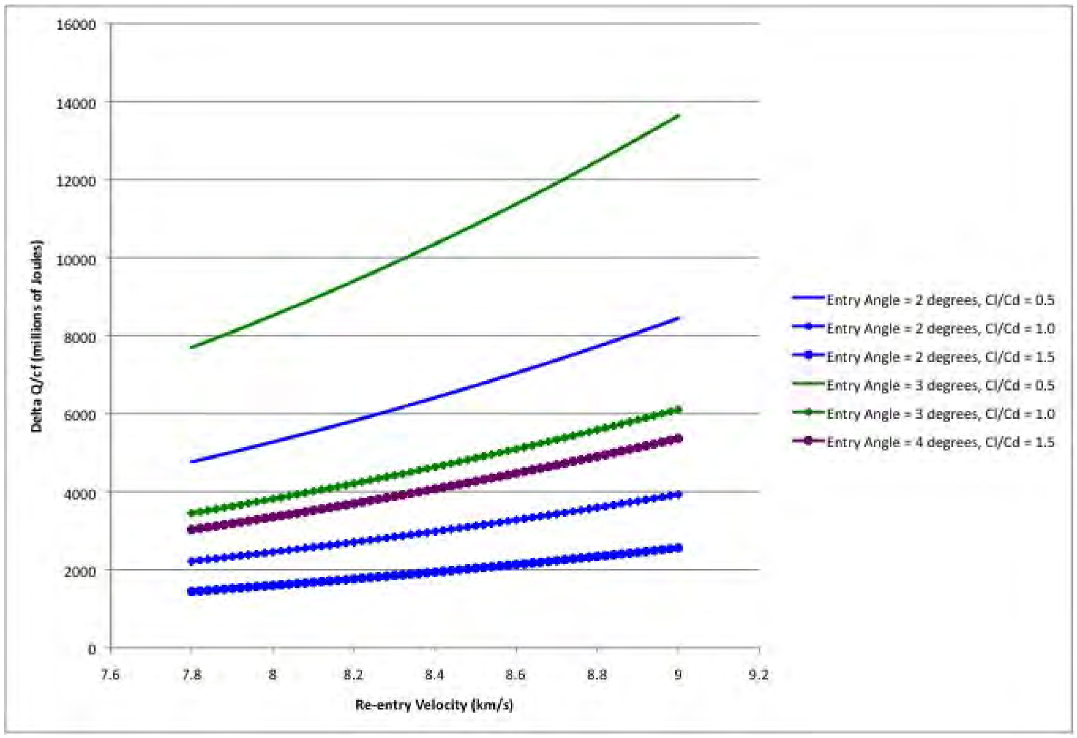


Figure 38: Closer Look at Lower Heating Loads (3)

As Figures 35-38 show, the re-entry angle plays a major role in the amount of heat the spacecraft will experience. Using a larger L/D ratio can mitigate this, but the larger re-entry velocities (larger radius of apogees) will need a significant amount of thermal protection, as these heating loads are similar to those of re-entry vehicles used today. For example, the Apollo missions expected to see a heating load of approximately 10^7 J/kg (Regan, 1984:436). This compares with the worst case of 7×10^6 J/kg and best case of 9×10^5 J/kg for the analysis above.

If the maximum re-entry angle can be maintained at a small angle, the amount of heating is significantly reduced. The higher L/D ratios can also be used to reduce the amount of heat experienced by the vehicle on average 53.4% for L/D of 1.0 and 69.7% for L/D of 1.5. However, the main way to reduce heat experienced by the vehicle will utilize an active heat control system, such as ablation or heat sinks. The design of the vehicle can significantly reduce the amount of heat experienced at the surface of the vehicle by using shocks to keep most of the heat away from the vehicle and only allow a small percentage of the total heat experienced during the re-entry event to get to the spacecraft. This heat could then be dissipated back out to space as the spacecraft progresses through the rest of its orbit. Further analysis will need to be completed in order to determine the type and size of heat control system needed.

Conclusion

Chapter Overview

This chapter will discuss the conclusions of the author formed from the analysis. These conclusions will then be used to discuss why this particular thesis is significant and provide recommendations for future actions for the computer code to provide more detailed analysis. Recommendations for future research will also be provided based on the conclusions of this thesis.

Conclusions of Research

Throughout the analysis of this thesis it has been shown using aerodynamic forces and skipping off the atmosphere can significantly improve the time on orbit when compared with a spacecraft not using these forces. Analysis has shown the addition of a thruster does not necessarily provide great improvement for spacecraft with initial near circular orbits at low perigee altitudes. These orbits also do not necessarily provide the improvements needed for a redesign of current spacecraft systems.

For the more eccentric orbits at beginning of life the thruster does provide a significant improvement for time on orbit when compared with the non-thruster satellites with and without waverider designs. An important aspect to note is the quick depletion of fuel for the thruster. As the orbit became more elliptical the thruster quickly transitioned to a constantly on device. This resulted in the fuel being used in the first few orbits. Further analysis should be completed in order to determine if removing the fuel and weight of the thruster would result in a better improvement in time on orbit due to the reduced weight than that of the increase from the thruster.

Heating analysis also shows the spacecraft can survive repeated skip re-entry events as long as an active heat control system is used and is able to dump the heat acquired during the events. Using higher L/D ratios would not only provide for a longer time on orbit but also decrease the heat load experienced by the spacecraft during a skip across the atmosphere at perigee. This higher L/D ratio would also allow for a greater range in re-entry angle the spacecraft could enter at with the same heat load for a spacecraft with a lower L/D ratio. Although initial analysis shows the spacecraft can survive these repeated skip re-entries, a large degree of further analysis needs to be completed in order to determine the optimum design of the vehicle for heating.

Significance of Research

A spacecraft utilizing the waverider design could greatly increase the resolution of an ORS satellite. Assuming a responsive launch capability can be developed a waverider spacecraft could be launched into an eccentric orbit with a low perigee and still maintain a short revisit time while greatly increasing the resolution capabilities. This waverider spacecraft would re-enter in a much shorter timeframe than current satellites, most likely under a month. However, the waverider design and heat control system would provide the added benefit of survivability during the final re-entry event. Further analysis and larger time steps during non-perigee transitions of the orbit analysis for the code could provide increased knowledge on the largest attainable orbit lifetime.

This type of vehicle can also lay the foundation for SR type missions. Although many other technological advancements are needed to make SR missions feasible the waverider spacecraft design and larger elliptical orbits could greatly increase the number of reachable locations. This type of orbit also allows the spacecraft to drop the collected debris off during a perigee event and potentially increases the amount of space debris collection. As stated earlier many other advancements in debris collection and debris de-orbiting technology are needed, but this analysis does provide groundwork for future debris removal options.

Recommendations for Action

The next step in furthering the analysis of this paper is to conduct more simulations for larger eccentricities and larger initial radiuses of perigee. The low number of orbits even with the thruster and aerodynamic forces for the 80 km perigee orbit shows future analysis should concentrate on larger perigees. This analysis would allow a determination of the optimum radius of perigee for contemplating a spacecraft using aerodynamic forces versus a traditional satellite. Eventually the eccentricity of the orbit will increase the orbital period to a point where a traditional satellite in a near circular larger semi-major axis is more appropriate for the mission.

An extensive analysis into the design and expected heating loads will need to be conducted in order to move this spacecraft design past the drawing board. The potentially high and repeatability of the heating loads this type of mission would experience makes heating management the driving factor of the mission. The risk associated with this heating analysis could be alleviated by borrowing practices from past and present missions such as the Apollo, Space Transportation System, and X-37B missions.

Recommendations for Future Research

The X-37B would also provide insight into the accuracy of the model by conducting its own high eccentricity, low perigee orbits and measuring the aerodynamic loads experienced. This would require an accurate model of the X-37B.

The computer model code can also be improved upon by creating algorithms for measuring the flight path angle, angle of attack, re-entry angle, and roll angle in real time. This would allow for a more dynamic modeling of the aerodynamic forces and greatly increase the accuracy of the model. It would also allow the user to better choose the type of conditions associated with the perigee events (re-entry angle, wet area, etc.). A real time modeling of the heating loads could also be incorporated in order to have the heating analysis for any simulation on hand right after the simulation run. The incorporation of the different angles would also allow for the user to create more complex models that use many surface areas of different sizes.

References

"Airbus A380 Dimensions and Key Data." Excerpt from unpublished article. http://www.airbus.com/aircraftfamilies/passengeraircraft/a380family/a380-800/specifications. 21 Jan 2012

Anderson, John D Jr. *Hypersonic and High-Temperature Gas Dynamics*. Reston VA: American Institute of Aeronautics and Astronautics, Inc., 2nd Edition, 2006.

Bate, Roger R. and others. *Fundamentals of Astrodynamics*. New York: Dover Publications, Inc, 1971.

Bettinger, Robert A. *Spacecraft Demand Tasking and Skip Entry Responsive Maneuvers.* MS thesis, AFIT/GAE/ENY/11-J03. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 2011

Carter, R. T., Jandir, P.S., and M. E. Kress. *"Estimating the Drag Coefficients of Meteorites for All Mach,"* 40th Lunar and Planetary Science Conference. RS5-2007-7004. San Jose CA, 2009.

Hall, Timothy S. *Orbit Maneuver for Responsive Coverage Using Electric Propulsion.* MS thesis, AFIT/GSS/ENY/10-M04. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 2010

Hicks, Kerry D. *Introduction to Astrodynamic Re-entry*. Air Force Institute of Technology (AU), Wright Patterson AFB OH, September 2009.

Jolley, Patrick R. and Stephen A. Whitmore. "Aerodynamic and Propulsion Assisted Maneuvering for Orbital Transfer Vehicles," 5th Responsive Space Conference. RS5-2007-7004. Los Angeles CA, 2007.

Malik, Tariq. "Debris from Space Collision Poses Threat to Other Satellites." Space.com. http://www.space.com/5540-debris-space-collision-poses-threat-satellites.html. 12 February 2009.

National Aeronautic and Space Administration. *US Standard Atmosphere Model, 1976.* NASA-TM-X-7433b. Washington D.C., 1976.

Pienkowski, John P. *Analysis of the Aerodynamic Orbital Transfer Capabilities of a Winged Re-entry Vehicle.* MS thesis. Naval Postgraduate School, Monterey CA, September 2002

Regan, Frank J. *Re-Entry Vehicle Dynamics.* Washington, DC: American Institute of Aeronautics and Astronautics, Inc., 1984.

Saltzman, Edwin J., Wang, K. Charles and Kenneth W. Iliff. *Flight-Determined Subsonic Lift and Drag Characteristics of Seven Lifting-Body and Wing-Body Reentry Vehicle Configurations With Truncated Bases.* Reno, NV: American Institute of Aeronautics and Astronautics, Inc., 37th AIAA Aerospace Sciences Meeting and Exhibit, 1999.

Vallado, David A. *Fundamentals of Astrodynamics and Applications*. El Segundo: Microcosm Press, 2004.

"X-51A Waverider." Excerpt from unpublished article. http://www.af.mil/information/factsheets/factsheet.asp?id=17986. 23 March 2011

Appendix A: Space Plane Requirement

Requirements Matrix for Mark II, III and IV
(Desired for Mark I)

Requirement	***Threshold***	***Objective***
Sortie Utilization Rates		
Peacetime sustained	0.10 sortie/day	0.20 sortie/day
War/exercise sustained (30 days)	0.33 sortie/day	0.50 sortie/day
War/exercise surge (7 days)	0.50 sortie/day	1.00 sortie/day
Turn Times		
Emergency war or peace	8 hours	2 hours
MOB peacetime sustained	2 days	1 day
MOB war/exercise sustained (30 days)	18 hours	12 hours
MOB war/exercise surge (7 days)	12 hours	8 hours
DOL peacetime sustained	3 days	1 day
DOL war/exercise sustained (30 days)	24 hours	12 hours
DOL war/exercise surge (7 days)	18 hours	8 hours
System Availability		
Mission capable rate	80 percent	95 percent
Flight and Ground Environments		
Visibility	0 ft	0 ft
Ceiling	0 ft	0 ft
Crosswind component	25 knots	35 knots
Total wind	40 knots	50 knots
Icing	light rime icing	moderate rime icing
Absolute humidity	30 gms/m3	45 gms/m3
Upper level winds	95th percentile shear	all shear conditions

Outside temperature	-20 to 100F	-45 to 120F
Precipitation	light	moderate
Space Environment		
Radiation level	TBD	TBD
Flight Safety		
Risk to friendly population	< 1 x 10-6	< 1 x 10-7
Flight Segment loss	< 1 loss /2000 sorties	< 1 loss/5000 sorties
Reliability	0.9995	0.9998
Cross Range		
Unrestricted pop-up cross range	600 NM	1200 NM
CONUS pop-up cross range	400 NM	600 NM
Orbital cross range	1200 NM	2400 NM
"Pop-up" Range		
CONUS pop-up range	1600 NM	1200 NM
Ferry range minimum	2000 NM	worldwide
On-orbit Maneuver		
Excess V (at expense of payload)	300 fps	600 fps
Pointing accuracy	15 milliradians	10 milliradians
Mission Duration		
On-orbit time	24 hours	72 hours
Emergency extension on-orbit	12 hours	24 hours
Orbital Impact		
Survival impact object size	0.1-cm diameter	1-cm diameter
Survival impact object mass	TBD	TBD
Survival impact velocity	TBD	TBD
Alert Hold		
Hold Mission Capable	15 days	30 days

Mission Capable to Alert 2-hour Status	4 hours	2 hours
Hold Alert 2-hour Status	3 days	7 days
Alert 2-hour to Alert 15-minute Status	1 hour 45 minutes	30 minutes
Hold Alert 15-minute Status	12 hours	24 hours
Alert 15 Minute to Launch	15 minutes	5 minutes
Design Life		
Primary Structure	250 sorties	500 sorties
Time between major overhauls	100 sorties	250 sorties
Engine life	100 sorties	250 sorties
Time between engine overhauls	50 sorties	100 sorties
Subsystem life	100 sorties	250 sorties
Take-off and Landing		
Runway size	10,000 ft x 150 ft	8000 ft x 150 ft
Runway load bearing	S65	S45
Vertical landing accuracy	50 ft	25 ft
Payload Container		
Container change-out	1 hour	30 minutes
Crew Station Environment (if rqd)		
Life support duration	24 hours	72 hours
Emergency extension on-orbit	12 hours	24 hours
Crew Escape (if rqd)		
Escape capability	subsonic	full envelope
Maintenance and Support		
Maintenance work hours/sortie	100 hours	50 hours
R&R engine	8 hours	4 hours

Appendix B: MatLab Script Files

<u>GaussianVOPMain.m</u>

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  MAIN routine for Gaussian VOP Equations
%  INPUT: Initial COEs
%  OUPUT: COEs vs time, Coverage Percentage,
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all
clear all
clc
format long

%% Initialize variables
close all
clear all
clc
format long
wgs84data
global  MU RE J2  EEsqrd RadPerDay TwoPI outctr ...
    Fsdragoutstep Frdragoutstep Fwdragoutstep ...
    Fsliftoutstep Frliftoutstep Fwliftoutstep ...
    Fsthrustoutstep Frthrustoutstep Fwthrustoutstep ...
    tavststep rpdeloutstep mass abegin aend initmassfuel initmass % declare global vars
so I don't have to pass them between subroutines

file=[6648.137, 0.005, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(0);...
      6527.09, 0.0075, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      6543.573, 0.01, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      6644.243, 0.25, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      6819.092, 0.5, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      7003.391, 0.075, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      7197.93, 0.1, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      7621.338, 0.15, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      8097.671, 0.2, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      8637.516, 0.25, deg2rad(40), deg2rad(0), deg2rad(0), deg2rad(180);...
      ];

 iteration=1;
while iteration <= 1


%% Begin Code

abegin=0;
aend=0;
tout(1,1)=0;
outctr=1;
initmass=2000;
initmassfuel=200;
targetlong=276*pi/180;
targetlat=40*pi/180;
MaxViewAngle=90*pi/180;
x_0 = [file(iteration,1), file(iteration,2), file(iteration,3), file(iteration,4),...
     file(iteration,5), file(iteration,6)]; % [a, e, inc, raan, arp, ma,]
rpinit=x_0(1,1)-x_0(1,1)*x_0(1,2);
t_0 = 0;  %intial time
dt=60;   %time step (seconds)
Yr = 2011;
Mon = 6;
D = 3;
H = 12;
M = 0;
S =  0;
JD=julianday(Yr,Mon,D,H,M,S);
```

```
x=[0,0,0,0,0,0];
t=0;
xstep=x_0;
x(1,:)=x_0;
t(1,1)=0;

tavst(outctr,1)=0;

Fsdragout(1,1)=0;
Frdragout(1,1)=0;
Fwdragout(1,1)=0;

Fsliftout(1,1)=0;
Frliftout(1,1)=0;
Fwliftout(1,1)=0;

Fsthrustout(1,1)=0;
Frthrustout(1,1)=0;
Fwthrustout(1,1)=0;

massout(1,1)=initmass;
rpdelout(1,1)=0;
mass=initmass;

%% Propogate
tstep=0;
while x(end,1)*(1-x(end,2))>6428 && tstep<25000 && x(end,2)>0.001;
options=odeset('MaxStep',5);
t_start=tstep;
t_end=tstep+dt;
[tout,xout]=ode45(@COEDeltasNew,[t_start,t_end],xstep,options);

% Compute output files
tavst(outctr,1)=tavststep;

Fsdragout(outctr+1,1)=Fsdragoutstep;
Frdragout(outctr+1,1)=Frdragoutstep;
Fwdragout(outctr+1,1)=Fwdragoutstep;

Fsliftout(outctr+1,1)=Fsliftoutstep;
Frliftout(outctr+1,1)=Frliftoutstep;
Fwliftout(outctr+1,1)=Fwliftoutstep;

Fsthrustout(outctr+1,1)=Fsthrustoutstep;
Frthrustout(outctr+1,1)=Frthrustoutstep;
Fwthrustout(outctr+1,1)=Fwthrustoutstep;

massout(outctr+1,1)=mass;
rpdelout(outctr+1,1)=rpdeloutstep;


t(outctr+1,1)=tout(end,1); % Plus one due to time of 0
x(outctr+1,:)=xout(end,:); % Plus one due to zero being x_0

% Step to next iteration
outctr=outctr+1;
tstep=tout(end,1);
xstep=xout(end,:);
end

%% Find ground track and coverage plots
gtctr=1;
while gtctr<=t_end/dt+1;

a=x(gtctr,1);
e=x(gtctr,2);
inc=x(gtctr,3);
raan=x(gtctr,4);
```

```
arp=x(gtctr,5);
ma=x(gtctr,6);


% Find Eccentric anomaly from mean anomaly
E=0;
Enew=revcheck(ma,2*pi);

while Enew-E>0.0001;
E=Enew;
mcheck=E-e*sin(E);
Enew=E+revcheck(ma,2*pi)-mcheck;
end
E=Enew;

%% Graph Ground Track and Determine if target is within view of satellite
% Find true anomaly from eccentric anomaly
ta=acos((cos(E)-e)/(1-e*cos(E)));
if E<=pi;
    ta=ta;
else
    ta=2*pi-ta;
end;

% compute R and V in pqw frame
[rpqw,vpqw]=RVpqw(a,e,ta);

% rotate into the ijk frame
[rijk,vijk]=RVijk(rpqw,vpqw,inc,raan,arp);

RIJK(gtctr,1)=rijk(1,1);
RIJK(gtctr,2)=rijk(2,1);
RIJK(gtctr,3)=rijk(3,1);
VIJK(gtctr,1)=vijk(1,1);
VIJK(gtctr,2)=vijk(2,1);
VIJK(gtctr,3)=vijk(3,1);

%Find updated Julian Date
if gtctr==1
    JD=JD;
else
    JD=JD+dt/86400;
end

%Find Greenwich Mean Sidereal Time
GMST=gstime(JD);

%%Determine if target is within view angle
%Update longitude with GMST
targetlongGMST=GMST+targetlong;

%Find target in IJK frame
targetIJK=[RE*sin(targetlongGMST)*cos(targetlat);...
           RE*sin(targetlongGMST)*cos(targetlat);...
           RE*sin(targetlat)];

%Find vector from satellite to target
SattoTargetIJK=targetIJK-rijk;

%Find angle between satellite to target vector and satellite vector
CurrentViewAngle(gtctr,1)=acos(dot(rijk,SattoTargetIJK)/(mag(rijk)*mag(SattoTargetIJK)));

%Determine if current view angle is less than max allotted view angle
if CurrentViewAngle(end,1)<MaxViewAngle
    viewcheckangle=1;
else
    viewcheckangle=0;
end;
```

```
%Determine if satellite's line of sight crosses through Earth
CrossedEarth = CrossThroughEarth(rijk,targetIJK);

%Find geodetic latitude and longitude directly below satellite
gclatr=sqrt(RIJK(gtctr,1)^2+RIJK(gtctr,2)^2);
alpha=acos(RIJK(gtctr,1)/gclatr);
if RIJK(gtctr,2)>=0;
    alpha=alpha;
else
    alpha=2*pi-alpha;
end;
gdlong=alpha-GMST;
gclat=atan2(RIJK(gtctr,3),gclatr);
gdlat=gclat;
gdlato=0;
rK=RIJK(gtctr,3);
while gdlat-gdlato>0.1;
gdlato=gdlat;
C=RE/(sqrt(1-EEsqrd*(sin(gdlat)^2)));
gdlat=atan2((rK+C*EEsqrd*sin(gdlat)),gclatr);
end;

%Determine if view checks pass
if CrossedEarth&&viewcheckangle==1
    viewcheck=1;
else
    viewcheck=0;
end;
% Convert to degrees
gdlat=gdlat*180/pi;
gdlong=gdlong*180/pi;

% Consolidate latitudes and longitudes and whether or not in view
gdlatout(gtctr,1)=gdlat;
gdlongout(gtctr,1)=gdlong;
viewcheckout(gtctr,1)=viewcheck;

gtctr=gtctr+1;
end

% Plot the Latitudes and Longitudes
%figure  % create a new figure
%plot(gdlongout,gdlatout);  %plot the Latitude vs Longitude
%title('Latitude and Longitude'); ylabel('deg'); xlabel('deg');

% Plot the coverage times
%figure
%plot(t,viewcheckout);  %plot the Coverage vs. time
%title('Satellite Coverage'); ylabel('Coverage'); xlabel('seconds');

%% Plot the COEs
%figure  % create a new figure
%subplot(3,1,1);  plot(t,x(:,1)); %plot the Semi-major axis vs time
%title('Semi-major Axis'); ylabel('km'); xlabel('time');
%subplot(3,1,2);  plot(t,x(:,2)); %plot the Eccentricity vs time
%title('Eccentricity'); ylabel('unitless'); xlabel('time');
%subplot(3,1,3);  plot(t,x(:,3)); %plot the Inclination vs time
%title('Inclination'); ylabel('rad'); xlabel('time');

%figure % create a new figure
%subplot(3,1,1);  plot(t,x(:,4)); %plot the RAAN vs time
%title('RAAN'); ylabel('rad'); xlabel('time');
%subplot(3,1,2);  plot(t,x(:,5)); %plot the Argument of Perigee vs time
%title('Argument of Perigee'); ylabel('rad'); xlabel('time');
%subplot(3,1,3);  plot(t,x(:,6)); %plot the Mean Anomaly vs time
%title('Mean Anomaly'); ylabel('rad'); xlabel('time');

%% Plot the Forces, mass, and rpdel vs time
```

```
ctr=1;
while ctr<=length(t)
Forcedrags(ctr,1)=Fsdragout(ctr,1)*massout(ctr,1);
Forcedragr(ctr,1)=Frdragout(ctr,1)*massout(ctr,1);
Forcedragw(ctr,1)=Fwdragout(ctr,1)*massout(ctr,1);

Forcelifts(ctr,1)=Fsliftout(ctr,1)*massout(ctr,1);
Forceliftr(ctr,1)=Frliftout(ctr,1)*massout(ctr,1);
Forceliftw(ctr,1)=Fwliftout(ctr,1)*massout(ctr,1);

Forcethrusts(ctr,1)=Fsthrustout(ctr,1)*massout(ctr,1);
Forcethrustr(ctr,1)=Frthrustout(ctr,1)*massout(ctr,1);
Forcethrustw(ctr,1)=Fwthrustout(ctr,1)*massout(ctr,1);
ctr=ctr+1;
end;

%figure  % create a new figure
%subplot(3,1,1);  plot(t,Forcedrags(:,1)); %plot the Drag Force in the s direction vs
time
%title('Drag S'); ylabel('N'); xlabel('time');
%subplot(3,1,2);  plot(t,Forcedragr(:,1)); %plot the Drag Force in the r direction vs
time
%title('Drag R'); ylabel('N'); xlabel('time');
%subplot(3,1,3);  plot(t,Forcedragw(:,1)); %plot the Drag Force in the w direction vs
time
%title('Drag W'); ylabel('N'); xlabel('time');

%figure  % create a new figure
%subplot(3,1,1);  plot(t,Forcelifts(:,1)); %plot the Lift Force in the s direction vs
time
%title('Lift S'); ylabel('N'); xlabel('time');
%subplot(3,1,2);  plot(t,Forceliftr(:,1)); %plot the Lift Force in the r direction vs
time
%title('Lift R'); ylabel('N'); xlabel('time');
%subplot(3,1,3);  plot(t,Forceliftw(:,1)); %plot the Lift Force in the w direction vs
time
%title('Lift W'); ylabel('N'); xlabel('time');

%figure  % create a new figure
%subplot(3,1,1);  plot(t,Forcethrusts(:,1)); %plot the Thrust Force in the s direction vs
time
%title('Thrust S'); ylabel('N'); xlabel('time');
%subplot(3,1,2);  plot(t,Forcethrustr(:,1)); %plot the Thrust Force in the r direction vs
time
%title('Thrust R'); ylabel('N'); xlabel('time');
%subplot(3,1,3);  plot(t,Forcethrustw(:,1)); %plot the Thrust Force in the w direction vs
time
%title('Thrust W'); ylabel('N'); xlabel('time');

%figure % create a new figure
%plot(t,rpdelout(:,1)); %plot the change in radius of perigee vs time
%title('Change in Radius of Perigee'); ylabel('km/s'); xlabel('time');

%figure % create a new figure
%plot(t,massout(:,1)); %plot the mass vs time
%title('Spacecraft Mass'); ylabel('kg'); xlabel('time');

%% Create Data for Plots
TimeMax(iteration,1) = max(t);
NumOrbits(iteration,1) = max(x(:,6))/(2*pi);
FuelMax(iteration,1) = initmass - min(massout);
eplot(iteration,1)=file(iteration,2);

%%
% Save to excel file
if iteration==1
filename='check file drag';
end
if iteration==2
```

```
filename='rp100_NoLift_NoThrust_e0075.xlsx';
end
if iteration==3
filename='rp100_NoLift_NoThrust_e01.xlsx';
end
if iteration==4
filename='rp100_NoLift_NoThrust_e025.xlsx';
end
if iteration==5
filename='rp100_NoLift_NoThrust_e05.xlsx';
end
if iteration==6
filename='rp100_NoLift_NoThrust_e075.xlsx';
end
if iteration==7
filename='rp100_NoLift_NoThrust_e1.xlsx';
end
if iteration==8
filename='rp100_NoLift_NoThrust_e15.xlsx';
end
if iteration==9
filename='rp100_NoLift_NoThrust_e2.xlsx';
end
if iteration==10
filename='rp100_NoLift_NoThrust_e25.xlsx';
end
xlswrite(filename, t, 'Sheet1', 'A2');
xlswrite(filename, x, 'Sheet1', 'B2');
%xlswrite(filename, Forcedrags, 'Sheet1', 'H2');
%xlswrite(filename, Forcedragr, 'Sheet1', 'I2');
%xlswrite(filename, Forcedragw, 'Sheet1', 'J2');
%xlswrite(filename, Forcelifts, 'Sheet1', 'K2');
%xlswrite(filename, Forceliftr, 'Sheet1', 'L2');
%xlswrite(filename, Forceliftw, 'Sheet1', 'M2');
%xlswrite(filename, Forcethrusts, 'Sheet1', 'N2');
%xlswrite(filename, Forcethrustr, 'Sheet1', 'O2');
%xlswrite(filename, Forcethrustw, 'Sheet1', 'P2');
%xlswrite(filename, massout, 'Sheet1', 'Q2');
%xlswrite(filename, gdlatout, 'Sheet1', 'R2');
%xlswrite(filename, gdlongout, 'Sheet1', 'S2');
%xlswrite(filename, viewcheckout, 'Sheet1', 'T2');

xlswrite('Plots.xlsx', eplot, 'Sheet1', 'A2');
xlswrite('Plots.xlsx', TimeMax, 'Sheet1', 'B2');
xlswrite('Plots.xlsx', NumOrbits, 'Sheet1', 'C2');
xlswrite('Plots.xlsx', FuelMax, 'Sheet1', 'D2');

%% Output to MatLab
%rpfinal=x(end,1)-x(end,1)*x(end,2);
%rpfinalalt=rpfinal-6378.137
%rpchange=rpfinal-rpinit
%massfinal=massout(end,1)
%deltaV=161*9.81*log(150/massfinal)
%mafinal=x(end,6);
%mafinal=revcheck(mafinal,2*pi);
%mafinal=rad2deg(mafinal)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
iteration=iteration+1;
end
```

COEDeltas.m

```matlab
function [xdot]=COEDeltasNew(t,x)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  MAIN routine for Gaussian VOP Calculations
%  INPUT: COEs
%         x(1)=a
%         x(2)=e
%         x(3)=inc
%        x(4)=raan
%         x(5)=arp
%        x(6)=ma
%
%  OUPUT: COE Deltas
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global MU J2 RE ...
    Fsdragoutstep Frdragoutstep Fwdragoutstep ...
    Fsliftoutstep Frliftoutstep Fwliftoutstep ...
    Fsthrustoutstep Frthrustoutstep Fwthrustoutstep ...
    tavststep rpdeloutstep mass abegin aend initmassfuel initmass % declare global vars

%%
% Get COEs from state vector
a=x(1);
e=x(2);
inc=x(3);
raan=x(4);
arp=x(5);
ma=x(6);
if ma>2*pi
    ma=ma-2*pi;
end

%%
% Define Spacecraft Parameters
fpa=2*pi/180;
Cd=1;
Cl=0;
A=2.67e-6; % Needs to be in km^2
Isp=200;
thrust=1/1000; % divide by 1000 to be in N-km/s^2 vs N-m/s^2
mdot=(thrust/(Isp*9.81/1000));

%%
% Find Eccentric anomaly from mean anomaly
E=0;
Enew=ma;
while Enew-E>0.0001;
E=Enew;
mcheck=E-e*sin(E);
Enew=E+ma-mcheck;
end;
E=Enew;

% Find true anomaly from eccentric anomaly
ta=acos((cos(E)-e)/(1-e*cos(E)));
if E<=pi;
    ta=ta;
else
    ta=2*pi-ta;
end;

% Find latitude periapsis
u=arp+ta;

% Find mean motion
n=sqrt(MU/a^3);
```

```
% Find semi-latus rectum
p=a*(1-e^2);

% Find r
r=p/(1+e*cos(ta));

% Find angular momentum
h=sqrt(r*(1+e*cos(ta))*MU);

%%
% compute R and V in pqw frame
[rpqw,vpqw]=RVpqw(x(1),x(2),ta);

% rotate into the ijk frame
[rijk,vijk]=RVijk(rpqw,vpqw,x(3),x(4),x(5));

%%
% Find needed control angle
if ta<deg2rad(315) && ta>=deg2rad(300);
    abegin=x(1);
else
end

if ta<deg2rad(45) && ta>=deg2rad(0);
    aend=x(1);
else
end

adelta=abegin-aend;

controlangle=adelta*MU/(4*(x(1)^3)*thrust*sqrt(1-(x(2)^2)));

controlangle=acos((cos(controlangle)-e)/(1-e*cos(controlangle)));

%%
% Thrust Profile
if ta>=pi-controlangle && ta<pi+controlangle;
    if initmassfuel-mdot*t>0
        %mass=initmass-mdot*t;
        Fsthrust=thrust/mass;
        Fsthrust=0;
    else
        Fsthrust=0;
    end
else
    Fsthrust=0;
end

if x(2)<0.001;
    Fsthrust=0;
else
    Fsthrust=Fsthrust;
end

if ta>=pi-controlangle && ta<pi+controlangle;
    if initmassfuel-mdot*t>0
        %mass=initmass-mdot*t;
        Frthrust=0;
    else
        Frthrust=0;
    end
else
    Frthrust=0;
end

if x(2)<0.001;
    Frthrust=0;
else
    Frthrust=Frthrust;
```

```
end

if ta>=pi-controlangle && ta<pi+controlangle;
    if initmassfuel-mdot*t>0
        %mass=initmass-mdot*t;
        Fwthrust=0;
    else
        Fwthrust=0;
    end
else
    Fwthrust=0;
end

if x(2)<0.001;
    Fwthrust=0;
else
    Fwthrust=Fwthrust;
end

%% Force due to drag
vrelsq=((n^2*a^2)/(1-e^2))*(1+(e^2)+(2*e*cos(ta)));
rho=atmos76(rijk);
Fsdrag=-(0.5*rho*(Cd*A/mass))*vrelsq*cos(fpa);
Frdrag=(0.5*rho*(Cd*A/mass))*vrelsq*sin(fpa);
Fwdrag=0;

%% Force due to lift
Fslift=(0.5*rho*(Cl*A/mass))*vrelsq*sin(fpa);
Frlift=(0.5*rho*(Cl*A/mass))*vrelsq*cos(fpa);
Fwlift=0;

%% Add all Forces
Fs=Fsdrag+Fslift+Fsthrust;
Fr=Frdrag+Frlift+Frthrust;
Fw=Fwdrag+Fwlift+Fwthrust;

%% Changes in COEs due to forces
% Compute change in semi-major axis
adel=((2*e*sin(ta))/(n*sqrt(1-e^2)))*Fr+((2*a*sqrt(1-e^2))/(n*r))*Fs;

% Compute change eccentricity
edel=((sqrt(1-e^2)*sin(ta))/(n*a))*Fr+((sqrt(1-e^2)/(n*a^2*e))*((a^2*(1-e^2)/r)-r))*Fs;

% Compute change in inclination
incdel=(r*cos(u)/(n*a^2*sqrt(1-e^2)))*Fw;

% Compute change in Right Ascension of the ascending node
raandel=(r*sin(u)/(n*a^2*sqrt(1-e^2)*sin(inc)))*Fw;

% Compute change in argument of perigee
arpdel=(-sqrt(1-e^2)*cos(ta)/(n*a*e))*Fr+((p/(e*h))*(sin(ta)*(1+(1/(1+e*cos(ta))))))*Fs-
(r*cot(inc)*sin(u)/(n*a^2*sqrt(1-e^2)))*Fw;

% Compute change in mean anomaly epoch
madelepoch=(1/(n*a^2*e))*((p*cos(ta)-2*e*r)*Fr-((p+r)*sin(ta))*Fs);

% Compute change in radius of perigee
rpdel=adel-e*adel-a*edel;

%% Changes in Right Ascension of the Ascending Node and Argument of Perigee due to J2 and
J3
% Change in Right Ascension of the Ascending Node
%raandelJ2=-1.5*n*J2*((RE/a)^2)*cos(inc)/((1-e^2)^2);

% Change in Argument of Perigee
%arpdelJ2=0.75*n*J2*((RE/a)^2)*(4-5*(sin(inc)^2))/((1-e^2)^2);

% Compute final deltas
madel=madelepoch+n;
```

```
%raandel=raandel; %+raandelJ2;
%arpdel=arpdel; %+arpdelJ2;

% Find xdot vector
xdot=[adel;edel;incdel;raandel;arpdel;madel];

% Compute variables for output files
tavststep=ta;

Fsdragoutstep=Fsdrag;
Frdragoutstep=Frdrag;
Fwdragoutstep=Fwdrag;

Fsliftoutstep=Fslift;
Frliftoutstep=Frlift;
Fwliftoutstep=Fwlift;

Fsthrustoutstep=Fsthrust;
Frthrustoutstep=Frthrust;
Fwthrustoutstep=Fwthrust;

rpdeloutstep=rpdel;
```

CrossThroughEarth.m

```
function CrossedEarth = CrossThroughEarth(rijk,targetIJK)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%-
%%
%%  Use           : CrossEarth = CrossThroughEarth(rijk,targetIJK)
%%
%%  This function determines whether the satellite is looking through Earth
%%  to see a target.
%%
%%  Algorithm     : Finds a unit vector from the target to the satellite and
%%          adds it to the target vector to create a displaced vector.
%%          This vector is then compared to the radius of Earth to
%%          determine if the satellite is looking through the Earth.
%%
%%  Author        : Lt Matt Goodson    USAFA/AFIT  505-803-4872  19 July 2011
%%
%%  Inputs        :
%%    rijk        - Satellite IJK vector                 km
%%    targetIJK   - Target IJK vector            km
%%
%%  OutPuts       :
%%    CrossEarth  - Check Parameter                  1 or a 0
%%                  1 for not crossing Earth
%%                  0 for crossing Earth
%%
%%  Constants     :
%%    RE          - radius of the Earth          km
%%
%%  Coupling      :
%%    mag      determines magnitude of a vector
%%
%%  References    :
%%    None
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%% Constants
global RE

%Find Lengths of vector components of vector from target to satellite
Ilength=rijk(1,1)-targetIJK(1,1);
Jlength=rijk(2,1)-targetIJK(2,1);
Klength=rijk(3,1)-targetIJK(3,1);
Gain=1;

%Find magnitude of new vector
mag=sqrt(Ilength^2+Jlength^2+Klength^2);

%Find unit vectors of new vector
Iunit=Ilength/mag;
Junit=Jlength/mag;
Kunit=Klength/mag;

%Find displaced vector
DisplacedIJK=[targetIJK(1,1)+Iunit*Gain;targetIJK(2,1)+Junit*Gain;targetIJK(3,1)+Kunit*Ga
in];

%Find magnitude of displaced vector
magDisplacedIJK=sqrt(DisplacedIJK(1,1)^2+DisplacedIJK(2,1)^2+DisplacedIJK(3,1)^2);

if magDisplacedIJK>RE
    CrossedEarth=1;
else
    CrossedEarth=0;
end
```

wgs84data.m

```
function wgs84data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                function wgs84data
%% This script provides global conversion factors and WGS 84 constants
%% that may be referenced by subsequent MatLab script files and functions.
%% Note these variables are case-specific and must be referenced as such.
%%
%% The function must be called once in either the MatLab workspace or from a
%% main program script or function. Any function requiring all or some of the
%% variables defined must be listed in a global statement as follows,
%%
%%  global Deg Rad MU RE OmegaEarth SidePerSol RadPerDay SecDay Flat EEsqrd ...
%%         EEarth J2 J3 J4 GMM GMS AU HalfPI TwoPI Zero_IE Small Undefined
%%
%% in part or in its entirety. Order is not relevent. Case is.
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%       Originally written by Capt Dave Vallado
%%       Modified and Extended for Ada by Dr Ron Lisowski
%%       Extended from DFASMath.adb by Thomas L. Yoder, LtCol, Spring 00
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global Deg Rad MU RE OmegaEarth SidePerSol RadPerDay SecDay Flat EEsqrd ...
       EEarth J2 J3 J4 GMM GMS AU HalfPI TwoPI Zero_IE Small Undefined

%%  Degrees and Radians
      Deg=180.0/pi;                       %% deg/rad
      Rad= pi/180.0;                      %% rad/deg

%%  Earth Characteristics from WGS 84
      MU=398600.5;                        %% km^3/sec^2
      RE=6378.137;                        %% km
      OmegaEarth=0.000072921151467;       %% rad/sec
      SidePerSol=1.00273790935;           %% Sidereal Days/Solar Day
      RadPerDay=6.30038809866574;         %% rad/day
      SecDay=86400.0;                     %% sec/day
      Flat=1.0/298.257223563;             %%
      EEsqrd=(2.0-Flat)*Flat;
      EEarth=sqrt(EEsqrd);
      J2= 0.00108263;
      J3=-0.00000254;
      J4=-0.00000161;

%%  Moon & Sun Characteristics from WGS 84
      GMM= 4902.774191985;                %% km^3/sec^2
      GMS= 1.32712438E11;                 %% km^3/sec^2
      AU=  149597870.0;                   %% km

%%  HALFPI,PI2           PI/2, & 2PI in various names
      HalfPI= pi/2.0;
      TwoPI= 2.0*pi;

      Zero_IE  = 0.015;                   %% Small number for incl & ecc purposes
      Small    = 1.0E-6;                  %% Small number used for tolerance purposes
      Undefined= 999999.1;
```

gstime.m

```
function Temp = gstime ( JD )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%-
%%
%%  Use           : Temp = gstime ( JD )
%%
%%  This function finds the Greenwich Sidereal time.  Notice just the integer
%%    part of the Julian Date is used for the Julian centuries calculation.
%%
%%  Algorithm     : Perform expansion calculation to obtain the answer
%%                  Check the answer for the correct quadrant and size
%%
%%  Author        : Capt Dave Vallado  USAFA/DFAS  719-472-4109  12 Feb 1989
%%  In Ada        : Dr Ron Lisowski    USAFA/DFAS  719-472-4110  17 May 1995
%%  In MatLab     : Dr Ron Lisowski    USAFA/DFAS  719-333-4109   2 Jul 2001
%%
%%  Inputs        :
%%    JD          - Julian Date                          days from 4713 B.C.
%%
%%  OutPuts       :
%%    GSTime      - Greenwich Sidereal Time              0 to 2Pi rad
%%
%%  Locals        :
%%    Temp        - Temporary variable for reals         rad
%%    Tu          - Julian Centuries from 1 Jan 2000
%%
%%  Constants     :
%%    TwoPi       - Defined in DFASMath package
%%    RadPerDay   - Rads Earth rotates in 1 Solar Day
%%
%%  Coupling      :
%%    revcheck      Simplified MOD function
%%
%%  References    :
%%    1989 Astronomical Almanac pg. B6
%%    Escobal       pg. 18 - 21
%%    Explanatory Supplement pg. 73-75
%%    Kaplan        pg. 330-332
%%    BMW           pg. 103-104
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%% Constants
global RadPerDay TwoPI


     Tu = ( fix(JD) + 0.5 - 2451545.0 ) / 36525.0;
     %%Temp= 1.753368559 + 628.3319705*Tu + 6.770708127E-06*Tu*Tu +
     Temp= 1.753368559 + 628.3319705*Tu + 6.770708127E-06*Tu^2 + RadPerDay*( (frac( JD )
- 0.5) );

%%%%%%%%%%%%%%%%%%%%%%- Check quadrants %%%%%%%%%%%%%%%%%%%%-
     Temp= revcheck (Temp, TwoPI);
%%     if Temp < 0.0
%%         Temp= Temp + TwoPI;
%%     end;
```

atmos76.m

*Note: All of code not shown, missing tabulated densities at end

```
function [RHO] = atmos76 ( R )
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%-
%%
%%  Use           : [RHO] = atmos76 ( R )
%%
%%  This function finds the atmospheric density at an altitude above an
%%    oblate earth given the position vector in the Geocentric Equatorial
%%    frame.  The position vector is in km's and the density is in kg/km**3.
%%    The density is based on the 1976 Standard Atmoshpere Model. It is
%%    computed using the data file and interpolation code obtained from
%%    Aerospace Mission Analysis with MatLab by David Eagle
%%
%%  Algorithm     : Find initial values
%%                  Loop to find the latitudes
%%                  Calculate the density through table look-up
%%
%%  Author(ATMOS) : Capt Dave Vallado  USAFA/DFAS  719-472-4109  20 Sep 1990
%%  In Ada        : Dr Ron Lisowski    USAFA/DFAS  719-472-4110  29 Jul 1997
%%  In MatLab     : Dr Ron Lisowski    USAFA/DFAS  719-333-4109  14 Nov 2001
%%  Use 1976 Data : Dr Ron Lisowski    USAFA/DFAS  719-333-4109  09 Dec 2003
%%
%%  Inputs        :
%%    R           - GEC Position vector                    km
%%
%%  Outputs       :
%%    RHO         - Density                                kg/km**3
%%
%%  Locals        :
%%    Rc          - Range of site w.r.t. earth center      km
%%    Height      - Height above earth w.r.t. site         km
%%    Alt         - Altitude above earth w.r.t. site       km
%%    OldDelta    - Previous value of DeltaLat             rad
%%    DeltaLat    - Diff between Delta and Geocentric lat  rad
%%    GeoDtLat    - Geodetic Latitude                     -Pi/2 to Pi/2 rad
%%    GeoCnLat    - Geocentric Latitude                   -Pi/2 to Pi/2 rad
%%    TwoFMinusF2 - 2*F - F squared
%%    OneMinusF2  - ( 1 - F ) squared
%%    Delta       - Declination angle of R in IJK system   rad
%%    Temp        - Diff between Geocentric/Geodetic lat   rad
%%    RSqrd       - Magnitude squared
%%    SinTemp     - Sine of Temp
%%    i           - index
%%
%%  Constants     :
%%    Flat        - Flatenning of the Earth                0.003352810664747352
%%    REarthKm    - Earth equatorial radius                6378.137
%%
%%  Coupling      :
%%    Mag         - Vector Magnitude
%%
%%  References    :
%%    Escobal       pg. 398-399 ( Conversion to Lat and Height )
%%    AMAM          pg. 237     (look-up 1976 Standard Atmosphere density)
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%-

     global RE Flat HalfPI
%%   CONSTANTS
      Ae = RE;


     %%%%%%%%%%%%%%%%%%%%%%%  Initialize values   %%%%%%%%%%%%%%%%%%%%-
     TwoFMinusF2= 2.0*Flat - Flat*Flat;
     OneMinusF2 = ( 1.0-Flat )^2;
     %%%%%%%%%%%%%%%%- Set up initial latitude value  %%%%%%%%%%%%%%%%

%%     Original Coding:
%%     Dellta= ArcTan( R(3) / SQRT( R(1)*R(1) + R(2)*R(2) ) );
%%     IF ABS( Dellta ) > Pi THEN
%%         Dellta= Dellta MOD Pi;
%%     End if;

%%    Modified Coding: Eliminates divide by zero RJL 10 Oct 95 %%%%%%-
     Dellta = asin (R(3) / mag(R));

     GeoCnLat= Dellta;
     OldDelta=  1.0;
     DeltaLat= 10.0;
     RSqrd  = mag(R)^2;

     %%%%%%- Iterate to find Geocentric and Geodetic Latitude  %%%%%%-
     iter= 1;
     while ( abs( OldDelta - DeltaLat ) > 0.00001 ) && ( iter < 10 ) ,
         OldDelta= DeltaLat;
         Rc     = Ae * sqrt( ( 1.0-TwoFMinusF2 ) / ...
                          ( 1.0-TwoFMinusF2*cos(GeoCnLat)*cos(GeoCnLat) ) );
     %%%%%%- If statment added to deal with exact polar cases  %%%%%%%%
     %%%%%%- RJL 29 Jul 97 %%%%%%%%
         if abs (GeoCnLat-HalfPI) < 0.000001 ,
            GeoDtLat= GeoCnLat;
         else
            GeoDtLat= atan( tan(GeoCnLat) / OneMinusF2 );
         end
         Temp   = GeoDtLat-GeoCnLat;
         SinTemp= sin( Temp );
         Height = sqrt( RSqrd-Rc*Rc*SinTemp*SinTemp ) - Rc*cos(Temp);
         DeltaLat= asin( Height*SinTemp / mag(R) );
         GeoCnLat= Dellta - DeltaLat;
         iter = iter + 1;
     end  %% While %%

     if iter >= 10 ,
         disp( 'ATMOS76 latitude iteration did NOT converge ' );
     end


     h= Height;

     %%%%%%%%%%%% Determine density based on altitude %%%%%%%%%%%%%%
     %%%%%%%%%%%%%%%  1976 Standard Atmosphere Data %%%%%%%%%%%%%%
% Check to see if data is in global
global ad76
if length(ad76) < 1
   atmos76dat;
end

% compute index and interpolation factor

if h>1000
  RHO = 0;
  %disp ('ATMOS76 - alt exceeds 1000 km - density set to 0.0');
else
  for i = 1:1:2001
    xi = 0.5 * (i - 1);
    xim1 = xi - 0.5;

     if (h <= xi)
       if (i == 1)
          xinfac = 0;
          index = 1;
       else
          xinfac = (h - xim1) / (xi - xim1);
          index = i - 1;
```

```
        end
        break;
      end
   end

   y1 = ad76(index);

   y2 = ad76(index + 1);

% atmospheric value

   RHO = y1 + xinfac * (y2 - y1);
end

% Local function to set up the data for ATMOS76

function atmos76dat
global ad76
ad76 = [ ...
.1225000E+10  ...
.1167273E+10  ...
.1111660E+10  ...
.1058104E+10  ...
.1006554E+10  ...
```

JulianDay.m

```
function JD=julianday(Yr,Mon,D,H,M,S)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%-
%%
%%  Use           : JD=julianday(Yr,Mon,D,H,M,S)
%%
%%  This function finds the Julian date given the Year, Month, Day, and Time.
%%    The Julian date is defined by each elapsed day since noon, 1 Jan 4713 BC.
%%    Julian dates are measured from this epoch at noon so astronomers
%%    observations may be performed on a single "day".  The year range is
%%    limited since machine routines for 365 days a year and leap years are
%%    valid in this range only.  This is due to the fact that leap years occur
%%    only in years divisible by 4 and centuries whose number is evenly
%%    divisible by 400. ( 1900 no, 2000 yes ... )
%%
%%  NOTE:  This Algorithm is taken from the 1988 Almanac for Computers,
%%    Published by the U.S. Naval Observatory.  The algorithm is good for dates
%%    between 1 Mar 1900 to 28 Feb 2100 since the last two terms (from the
%%    Almanac) are commented out.  Including the last two terms enables
%%    calculations between 1 Mar 1801, and 2100.
%%
%%  Algorithm     : Find the various terms of the expansion
%%                  Calculate the answer
%%
%%  Author        : Capt Dave Vallado     USAFA/DFAS  719-472-4109  12 Aug 1988
%%  In Ada        : Dr Ron Lisowski       USAFA/DFAS  719-472-4110  17 May 1995
%%  In MatLab     : LtCol Thomas L. Yoder USAFA/DFAS  719-333-4110  Spring 00
%%
%%  Inputs        :
%%    Yr          - Year                                 1900 .. 2100
%%    Mon         - Month                                   1 .. 12
%%    D           - Day                                     1 .. 28,29,30,31
%%    H           - Universal Time Hour                     0 .. 23
%%    M           - Universal Time Min                      0 .. 59
%%    Sec         - Universal Time Sec                    0.0 .. 59.999
%%
%%  Outputs       :
%%    JD          - Julian Date                           days from 4713 B.C.
%%
%%  Locals        :
%%    Term1       - Temporary Long_Float value
%%    Term2       - Temporary 32 bit INTEGER value
%%    Term3       - Temporary 32 bit INTEGER value
%%    UT          - Universal Time                            days
%%
%%  Constants     : None.
%%
%%  Coupling      : None.
%%
%%  References    :
%%    1988 Almanac for Computers  pg. B2
%%    Escobal       pg. 17-19
%%    Kaplan        pg. 329-330
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TERM1= 367.0 * Yr;
TERM2= fix( ( 7.0 * (Yr + fix((Mon+9.0)/12.0))) * 0.25);
TERM3= fix(275.0 * Mon / 9.0 );
UT= ( (S/60.0 + M ) / 60.0 + H ) / 24.0;

JD= (TERM1-TERM2+TERM3) + D + UT + 1721013.5;
%%   The following neglected term must be added for dates before 28 Feb 1900
%%      + 0.5*sign( (100*Yr + Mon) - 190002.5) + 0.5;
```

revcheck.m

```
function y = revcheck (x, modby)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Use          : y = revcheck (x, modby)
%%
%%    Accomplishes a modulus function of x by modby. While this does exactly
%%    the same operation as the MatLab mod function, it does it quicker
%%    because the overhead is minimal. This has shown to be significant when
%%    propagating COEs at small time steps over large time spans.
%%
%%  inputs:
%%           x -  argument (radians, degrees, etc.)
%%       modby -  value to mod by (TwoPi, 360, etc.)
%%
%%  outputs:
%%           y -  x modulus modby
%%
%%  Author        : Capt Dave Vallado  USAFA/DFAS  719-472-4109   12 Aug 1988
%%  In Ada        : Dr Ron Lisowski    USAFA/DFAS  719-472-4110    2 Jul 1997
%%  In MatLab     : LtCol Thomas Yoder USAFA/DFAS  719-333-4110   Spring 2001
%%
%%  Locals:  None.
%%
%%  Constants:  None.
%%
%%  Coupling:  None.
%%  References:  None.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y = x - modby * fix(x / modby);

if (y ~= 0 && sign(y) ~= sign(modby))
   y = y + modby;
end
```

RVpqw.m

```
function [rpqw,vpqw]=RVpqw(a,ecc,nu)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Use: [rpqw,vpqw]=RVpqw(a,ecc,nu)
%
% The function RVpqw computes R and V in the pqw frame
%
% Author: Scott Dahlke  USAFA/DFAS  719-333-4110
%
% Inputs:
%   a - semimajor axis (km)
%   ecc - eccentricity
%   nu - true anomaly (rad)
%
% Outputs:
%   rpqw - position in the pqw frame
%   vpqw - velocity in the pqw frame
%
% Globals: MU
%
% Constants: None
%
% Coupling: None
%
% References:
%   COE's to RV Lesson of Astro 201
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global MU

% compute the parameter p
p = a*(1-ecc^2);

% compute the position vector
rpqw = p/(1 + ecc*cos(nu))*[cos(nu);sin(nu);0];

% compute the velocity vector
vpqw = sqrt(MU/p)*[-sin(nu);(ecc+cos(nu));0];
```

RVijk.m

```
function [rijk,vijk]=RVijk(rpqw,vpqw,incl,raan,argp)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Use: [rijk,vijk]=RVijk(rpqw,vpqw,incl,raan,argp)
%
% The function RVijk rotates R and V in the pqw frame to
% the ijk frame
%
% Author: Scott Dahlke  USAFA/DFAS  719-333-4110
%
% Inputs:
%   rpqw - position in the pqw frame
%   vpqw - velocity in the pqw frame%
%   incl - inclination (rad)
%   raan - right ascension of the ascending node (rad)
%   argp - argument of perigee (rad)
%
% Outputs:
%   rijk - position in the pqw frame
%   vijk - velocity in the pqw frame
%
% Globals: MU
%
% Constants: None
%
% Coupling: axisrot
%
% References:
%   COE's to RV Lesson of Astro 201
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global MU

% rotate the position vector
rtemp1 = axisrot(rpqw,3,-argp);
rtemp2 = axisrot(rtemp1,1,-incl);
rijk   = axisrot(rtemp2,3,-raan);

% rotate the velocity vector
vtemp1 = axisrot(vpqw,3,-argp);
vtemp2 = axisrot(vtemp1,1,-incl);
vijk   = axisrot(vtemp2,3,-raan);
```

axisrot.m

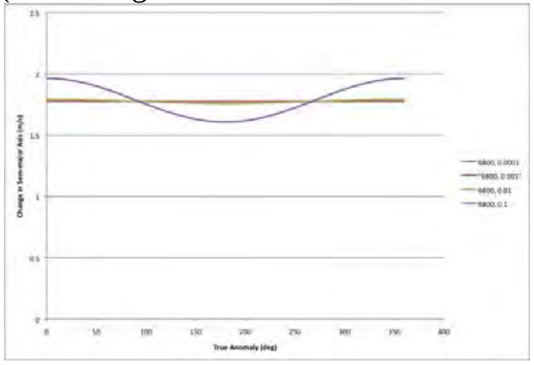
```
function B=axisrot(A,axis,alpha)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Use           : B=axisrot(A,axis,alpha)
%%
%% This function performs a rotation of angle ALPHA about a desired axis.
%%
%%   Author       : Dr. RON LISOWSKI, DFAS,      5 Jan 95
%%   In MatLab    : Thomas L. Yoder, LtC, USAFA, Spring 00
%%
%%   Input        :
%%     A          % Input vector                           Vector of dimension three
%%     axis       % desired axis for rotation:             1, 2 or 3
%%     alpha      % Angle of rotation                      radians
%%
%%   Output       :
%%     B          % Rotated Vector                         Vector of dimension three
%%
%%   Locals       : None.
%%
%%   Coupling     :
%%     mag        % Finds the magnitude of a vector
%%
%%   References   :
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

switch axis
case 1
   %rotate about the 1st axis
   B(1)=A(1);
   B(2)=A(2)*cos(alpha)+A(3)*sin(alpha);
   B(3)=-A(2)*sin(alpha)+A(3)*cos(alpha);
case 2
   % rotate about the 2nd axis
   B(1)=A(1)*cos(alpha)-A(3)*sin(alpha);
   B(2)=A(2);
   B(3)=A(1)*sin(alpha)+A(3)*cos(alpha);
case 3
   % rotate about the 3rd axis
   B(1)=A(1)*cos(alpha)+A(2)*sin(alpha);
   B(2)=-A(1)*sin(alpha)+A(2)*cos(alpha);
   B(3)=A(3);
otherwise
   disp('AxisRot axis number not 1, 2 or 3')
   B = A;
end
   % Ensure B is a column vector
   B = B(:);
```

mag.m

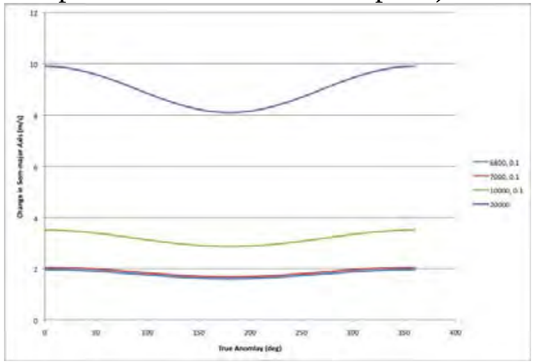
```
function B=mag(A)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Use           : B=mag(A)
%%
%%   Author       : Dr. RON LISOWSKI, DFAS,      5 Jan 95
%%   In MatLab    : Thomas L. Yoder, LtC, USAFA, Spring 00
%%
%% Overview: This function Calculates the Magnitude of a Vector
%%          Using this function in lieu of the general norm function
%%           may speed execution time in large data processing tasks
%%
%%
%%   Input        :
%%      A         - Input Vector
%%
%%   Output       :
%%      B         - Output magnitude float
%%
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[M,N]=size(A);
if M>N  % for column vector
   B=sqrt(A(1,1)^2+A(2,1)^2+A(3,1)^2);
else    % for row vector
   B=sqrt(A(1,1)^2+A(1,2)^2+A(1,3)^2);
end
```

Appendix C: Gaussian Variation of Parameters Graphs

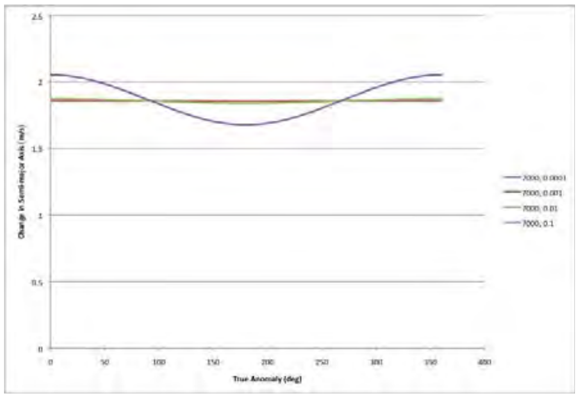
(Note: Original files can be found at L:\Research\Responsive Orbits\VOP Graphs)



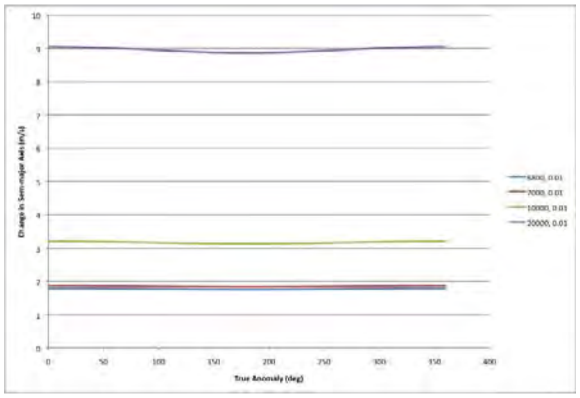
Change in Semi-major Axis Due to Changing Eccentricity and 6800 Semi-major Axis, S-direction



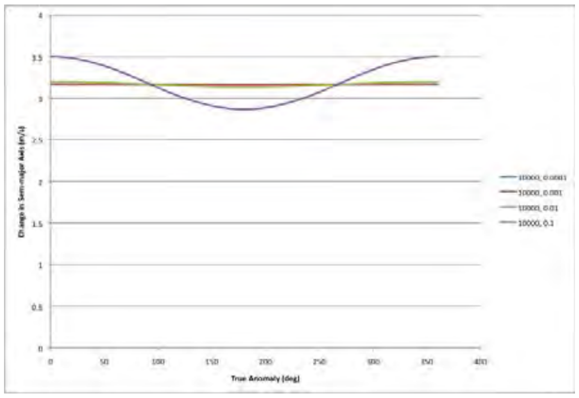
Change in Semi-major Axis Due to Changing Semi-major Axis and 0.1 Eccentricity, S-direction



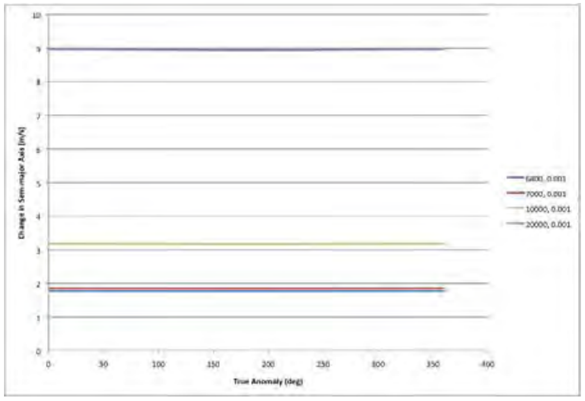
Change in Semi-major Axis Due to Changing Eccentricity and 7000 Semi-major Axis, S-direction



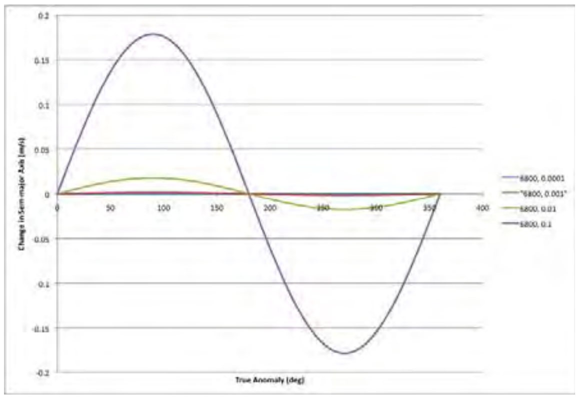
Change in Semi-major Axis Due to Changing Semi-major Axis and 0.01 Eccentricity, S-direction



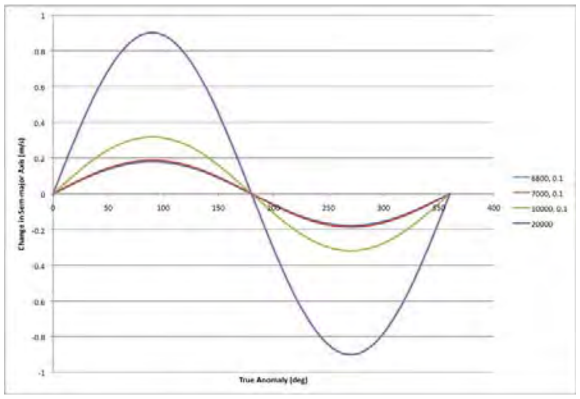
Change in Semi-major Axis Due to Changing Eccentricity and 10000 Semi-major Axis, S-direction



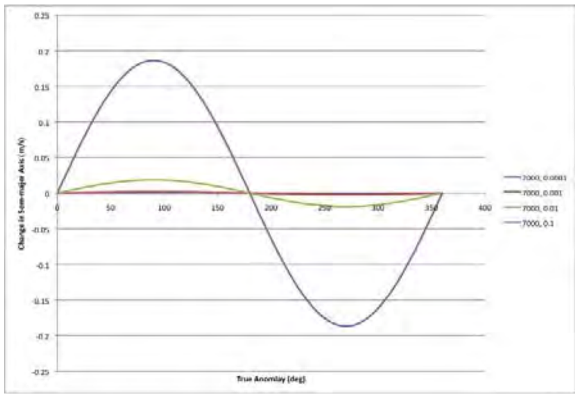
Change in Semi-major Axis Due to Changing Semi-major Axis and 0.001 Eccentricity, S-direction



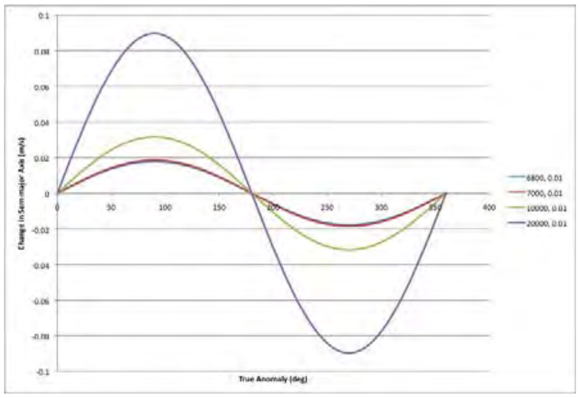
Change in Semi-major Axis Due to Changing Eccentricity and 6800 Semi-major Axis, R-direction



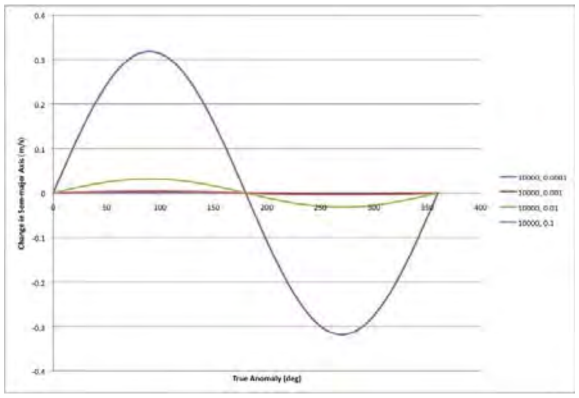
Change in Semi-major Axis Due to Changing Semi-major Axis and 0.1 Eccentricity, R-direction



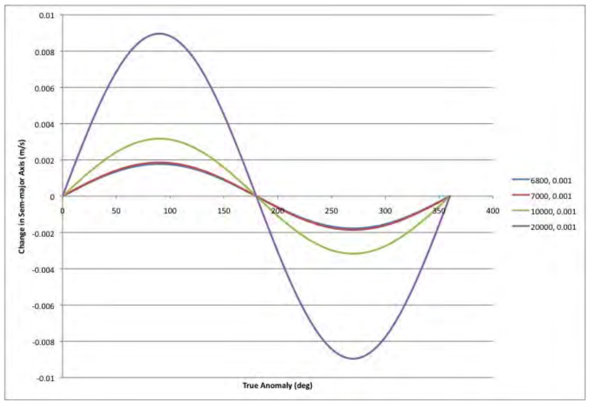
Change in Semi-major Axis Due to Changing Eccentricity and 7000 Semi-major Axis, R-direction



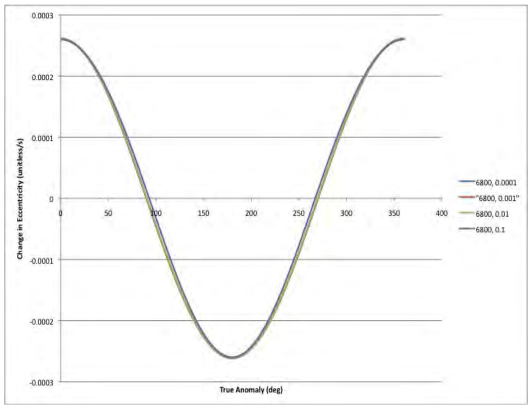
Change in Semi-major Axis Due to Changing Semi-major Axis and 0.01 Eccentricity, R-direction



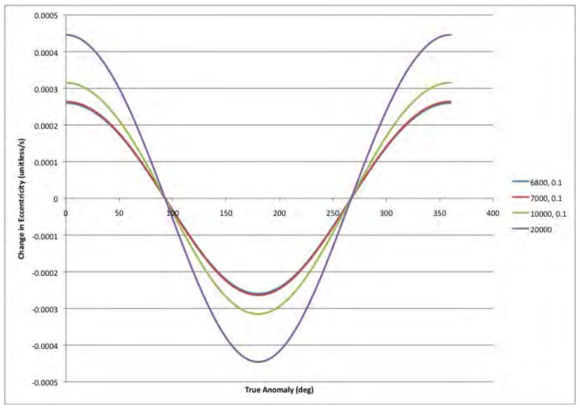
Change in Semi-major Axis Due to Changing Eccentricity and 10000 Semi-major Axis, R-direction



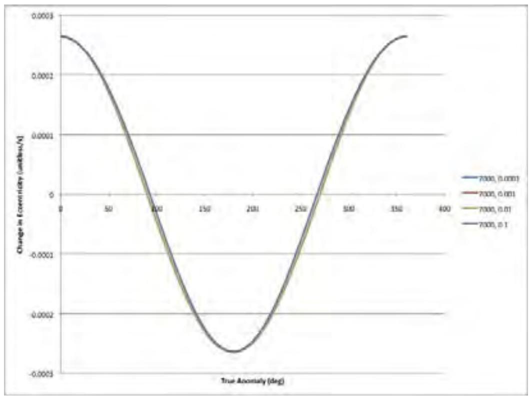
Change in Semi-major Axis Due to Changing Semi-major Axis and 0.001 Eccentricity, R-direction



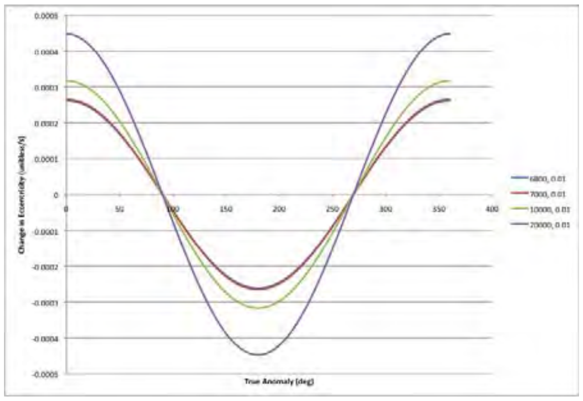
Change in Eccentricity Due to Changing Eccentricity and 6800 Semi-major Axis, S-direction



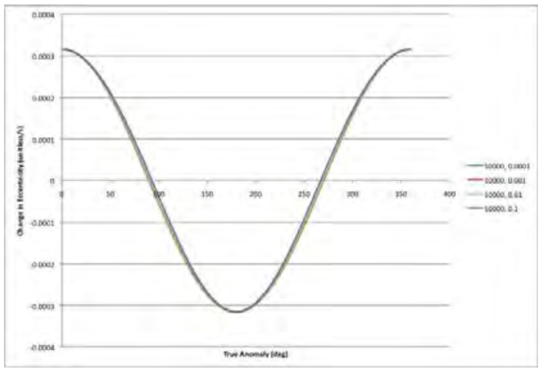
Change in Eccentricity Due to Changing Semi-major Axis and 0.1 Eccentricity, S-direction



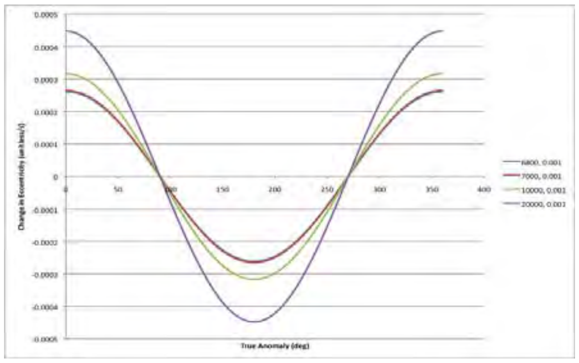
Change in Eccentricity Due to Changing Eccentricity and 7000 Semi-major Axis, S-direction



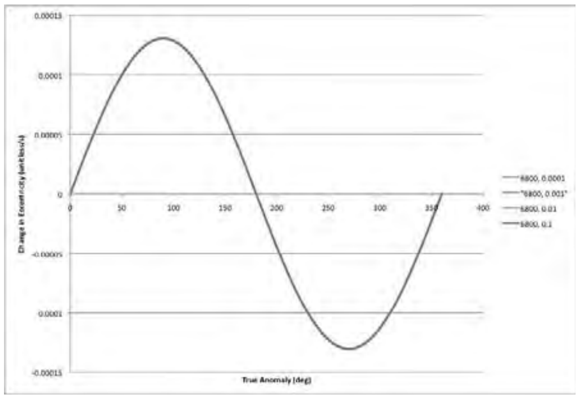
Change in Eccentricity Due to Changing Semi-major Axis and 0.01 Eccentricity, S-direction



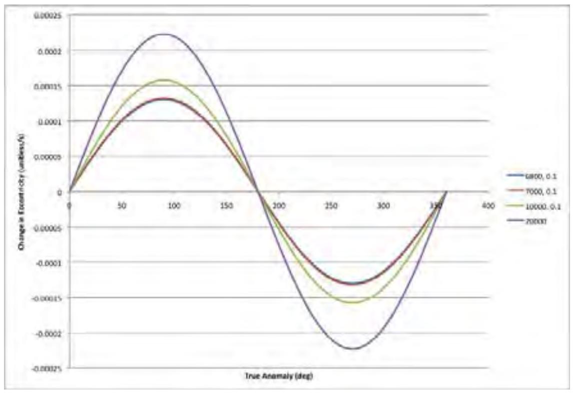
Change in Eccentricity Due to Changing Eccentricity and 10000 Semi-major Axis, S-direction



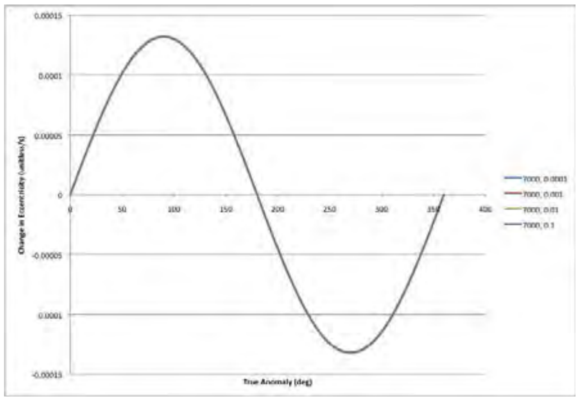
Change in Eccentricity Due to Changing Semi-major Axis and 0.001 Eccentricity, S-direction



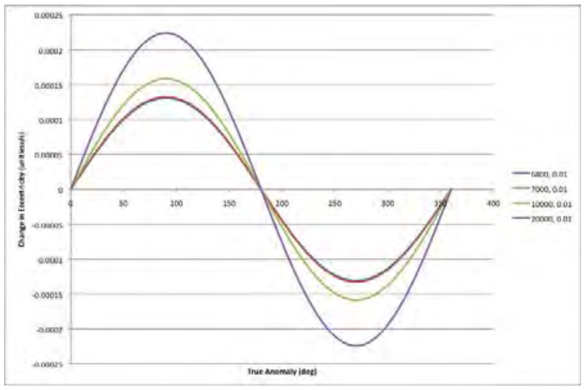
Change in Eccentricity Due to Changing Eccentricity and 6800 Semi-major Axis, R-direction



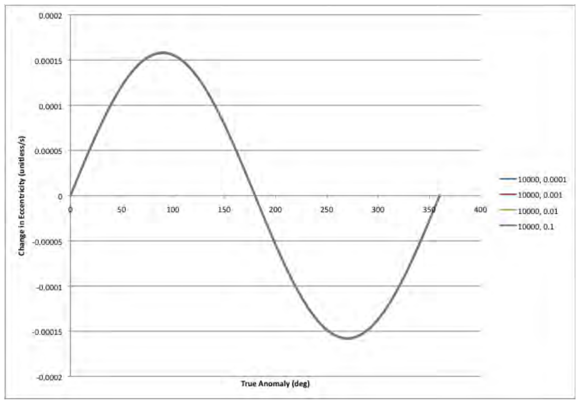
Change in Eccentricity Due to Changing Semi-major Axis and 0.1 Eccentricity, R-direction



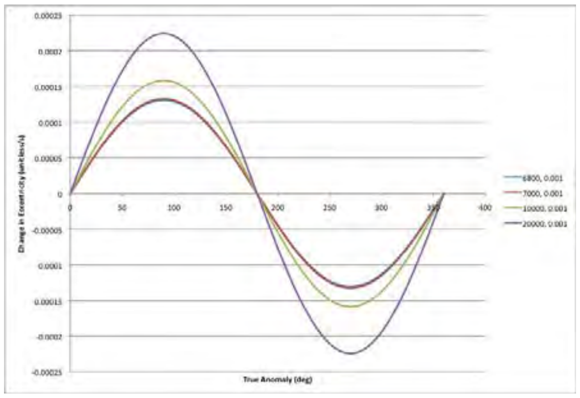
Change in Eccentricity Due to Changing Eccentricity and 7000 Semi-major Axis, R-direction



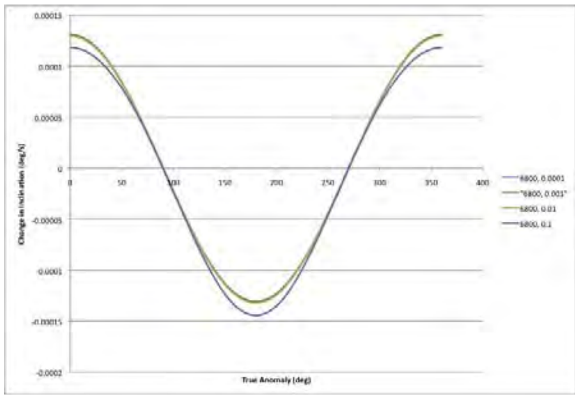
Change in Eccentricity Due to Changing Semi-major Axis and 0.01 Eccentricity, R-direction



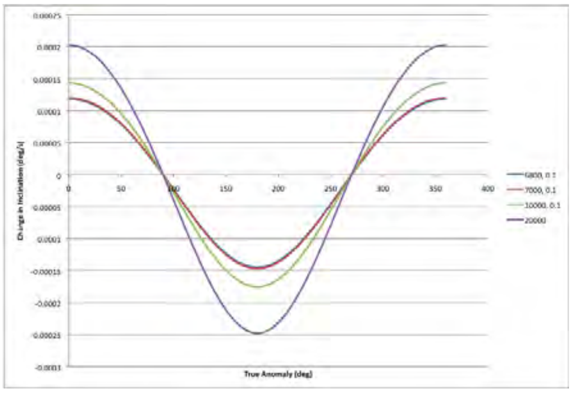
Change in Eccentricity Due to Changing Eccentricity and 10000 Semi-major Axis, R-direction



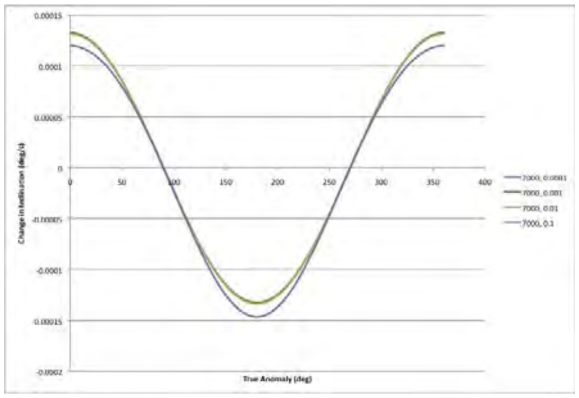
Change in Eccentricity Due to Changing Semi-major Axis and 0.001 Eccentricity, R-direction



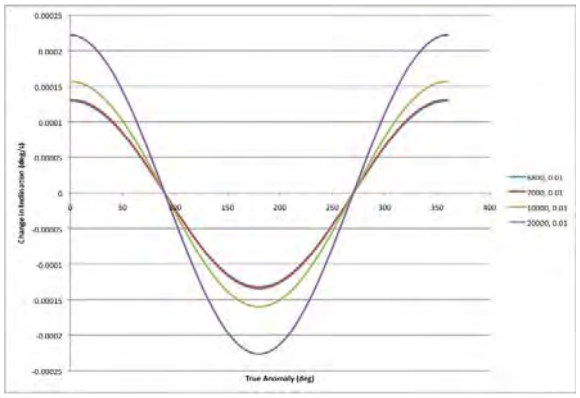
Change in Inclination Due to Changing Eccentricity and 6800 Semi-major Axis, W-direction



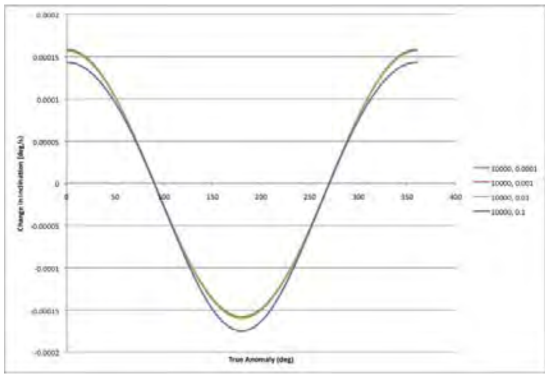
Change in Inclination Due to Changing Semi-major Axis and 0.1 Eccentricity, W-direction



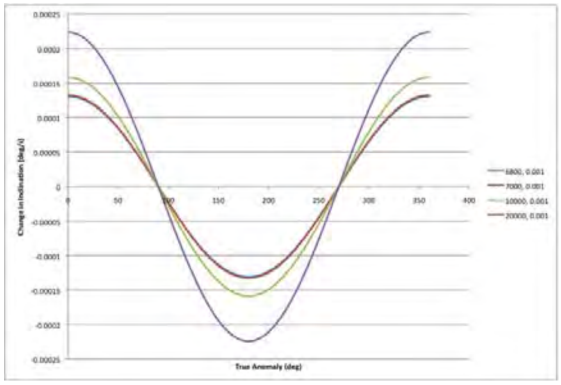
Change in Inclination Due to Changing Eccentricity and 7000 Semi-major Axis, W-direction



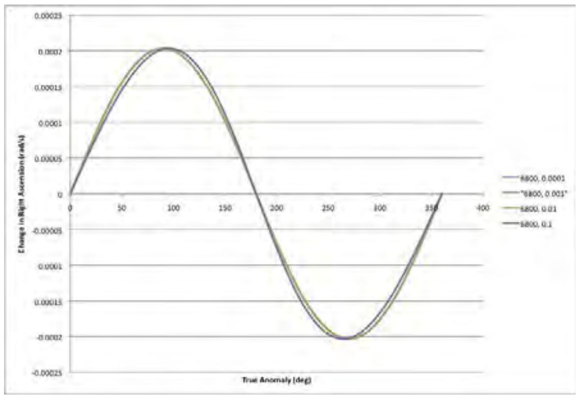
Change in Inclination Due to Changing Semi-major Axis and 0.01 Eccentricity, W-direction

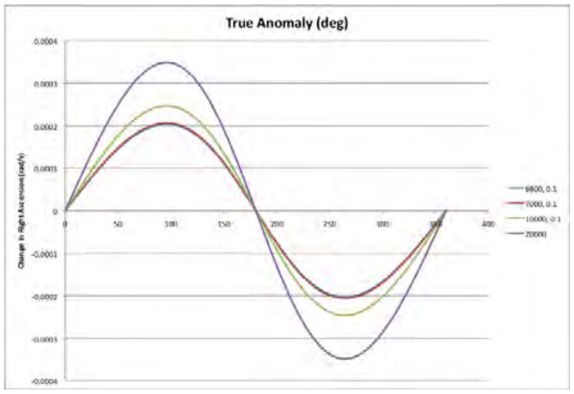


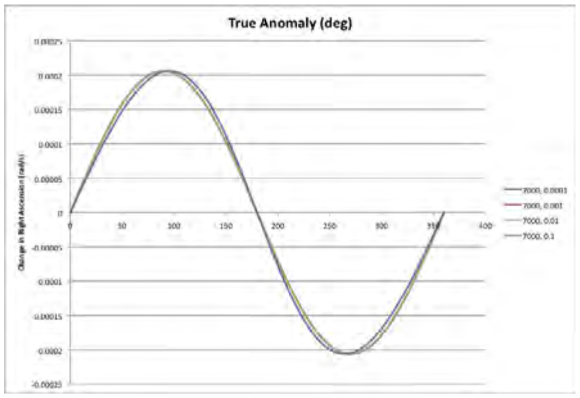
Change in Inclination Due to Changing Eccentricity and 10000 Semi-major Axis, W-direction

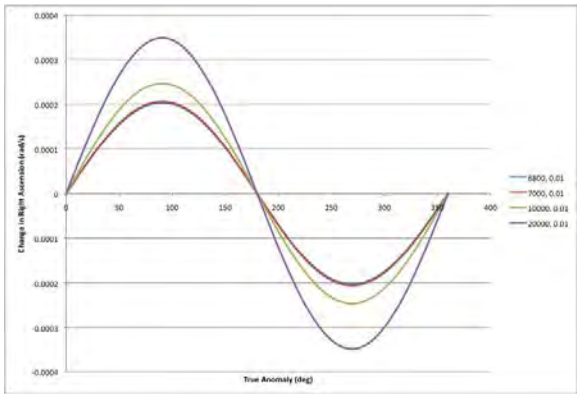


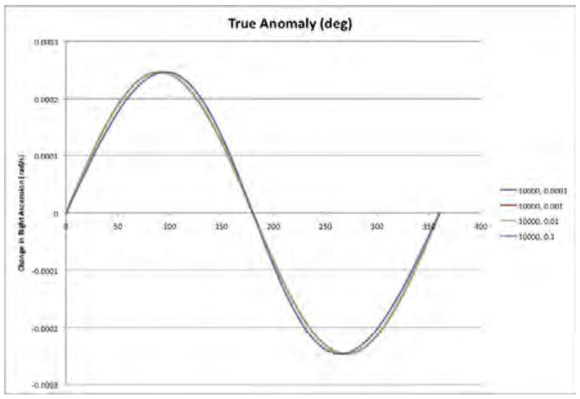
Change in Inclination Due to Changing Semi-major Axis and 0.001 Eccentricity, W-direction

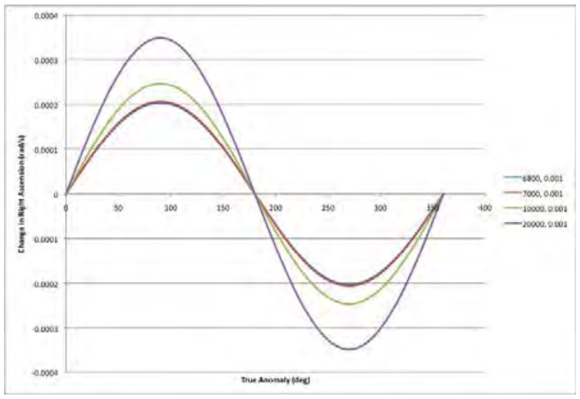

Change in Right Ascension Due to Changing Eccentricity and 6800 Semi-major Axis, W-direction

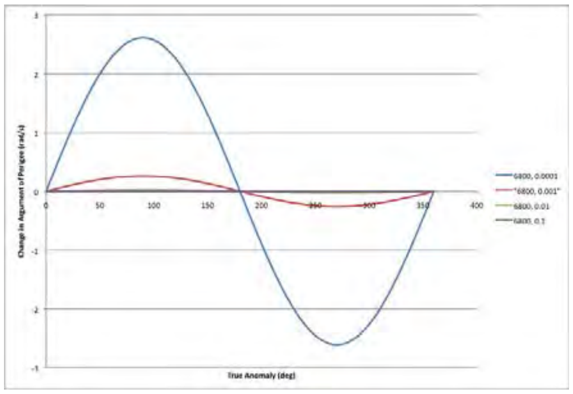

Change in Right Ascension Due to Changing Semi-major Axis and 0.1 Eccentricity, W-direction


Change in Right Ascension Due to Changing Eccentricity and 7000 Semi-major Axis, W-direction

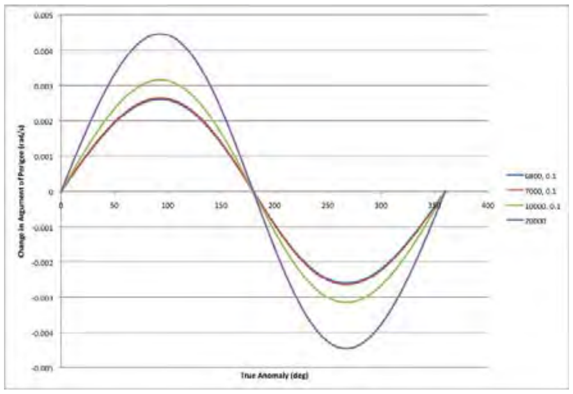

Change in Right Ascension Due to Changing Semi-major Axis and 0.01 Eccentricity, W-direction


Change in Right Ascension Due to Changing Eccentricity and 10000 Semi-major Axis, W-direction

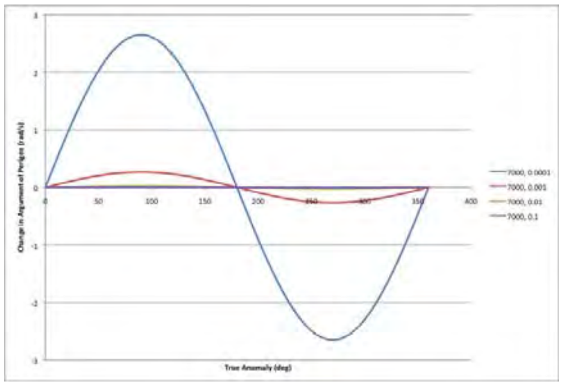

Change in Right Ascension Due to Changing Semi-major Axis and 0.001 Eccentricity, W-direction



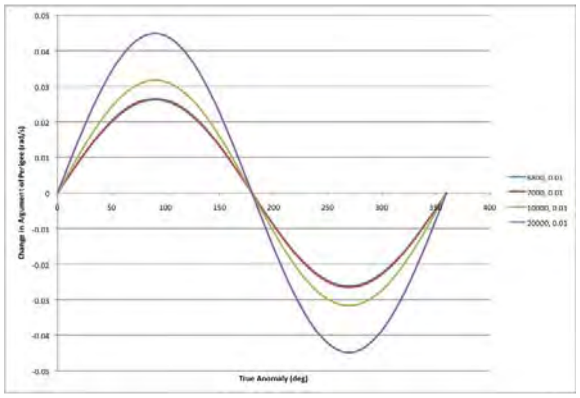
Change in Argument of Perigee Due to Changing Eccentricity and 6800 Semi-major Axis, S-direction



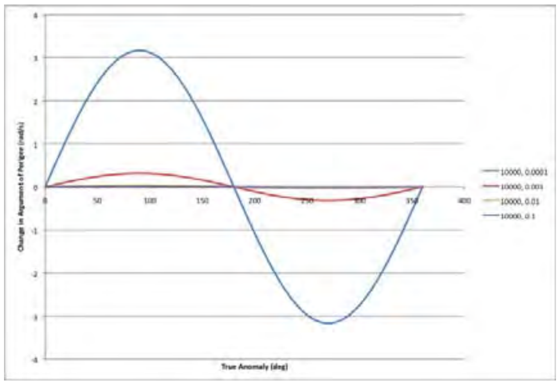
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.1 Eccentricity, S-direction



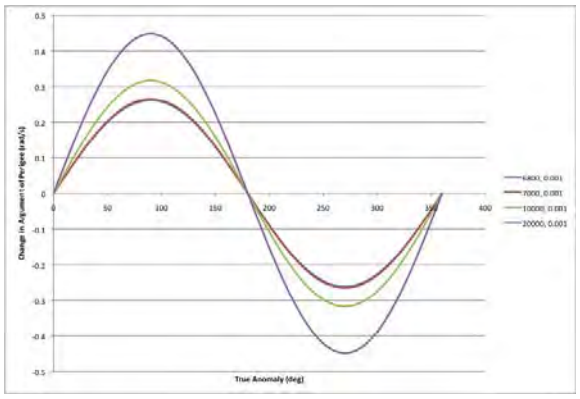
Change in Argument of Perigee Due to Changing Eccentricity and 7000 Semi-major Axis, S-direction



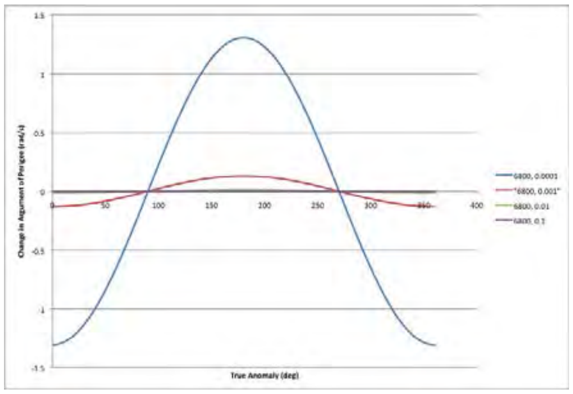
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.01 Eccentricity, S-direction



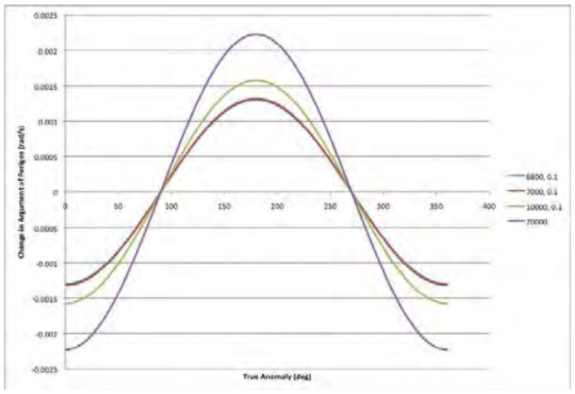
Change in Argument of Perigee Due to Changing Eccentricity and 10000 Semi-major Axis, S-direction



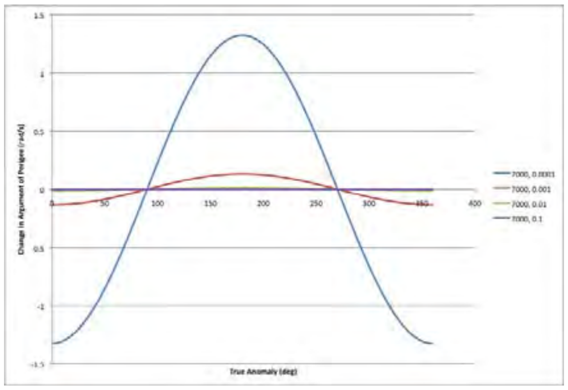
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.001 Eccentricity, S-direction



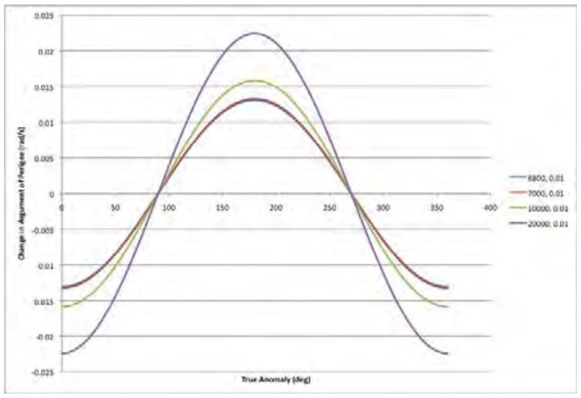
Change in Argument of Perigee Due to Changing Eccentricity and 6800 Semi-major Axis, R-direction



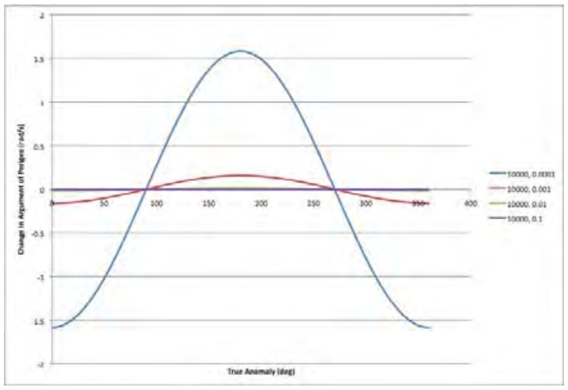
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.1 Eccentricity, R-direction



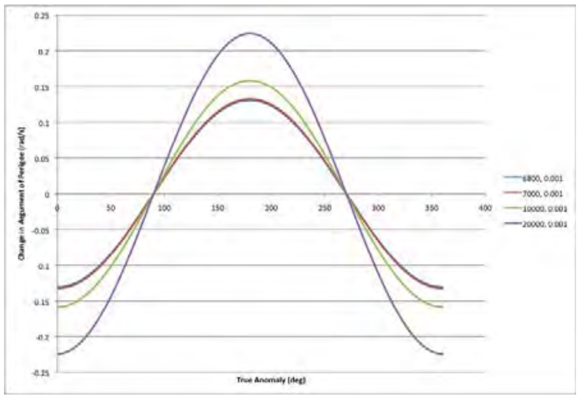
Change in Argument of Perigee Due to Changing Eccentricity and 7000 Semi-major Axis, R-direction



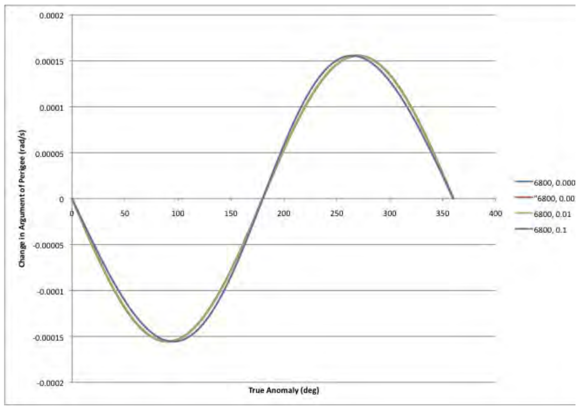
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.01 Eccentricity, R-direction



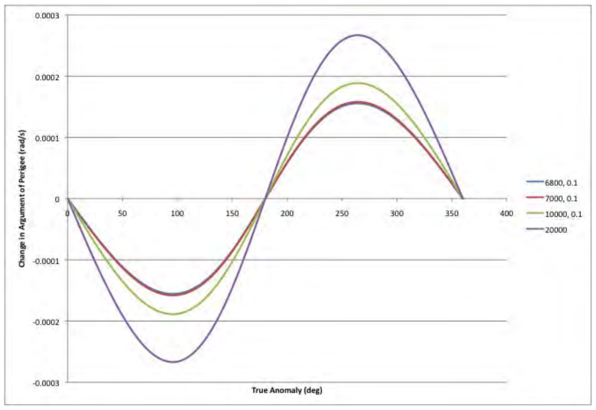
Change in Argument of Perigee Due to Changing Eccentricity and 10000 Semi-major Axis, R-direction



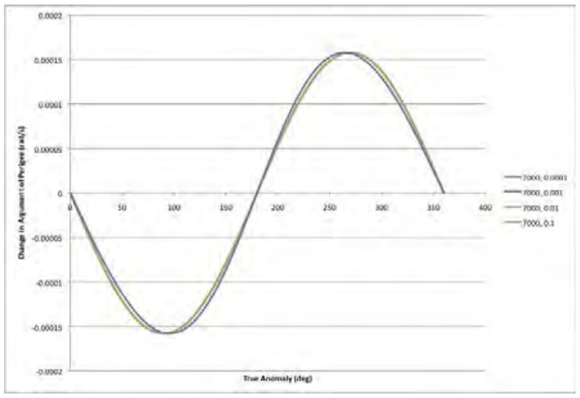
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.001 Eccentricity, R-direction



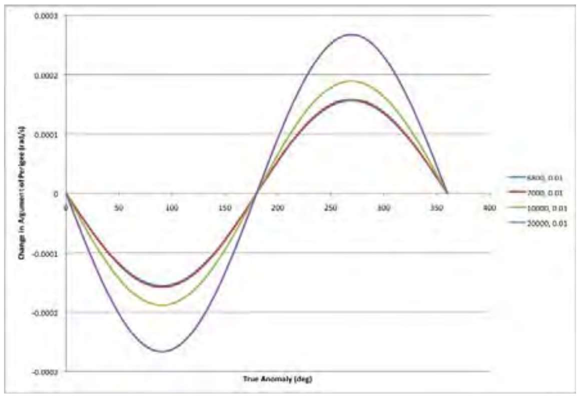
Change in Argument of Perigee Due to Changing Eccentricity and 6800 Semi-major Axis, W-direction



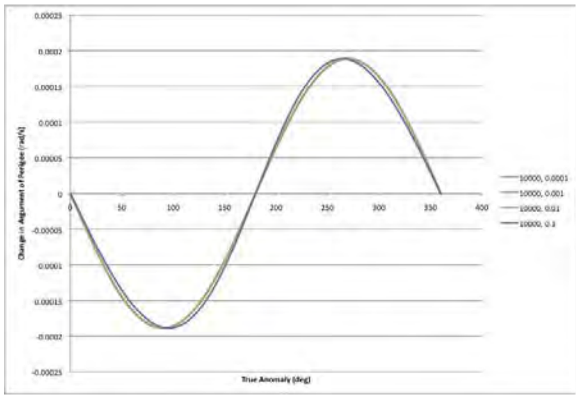
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.1 Eccentricity, W-direction



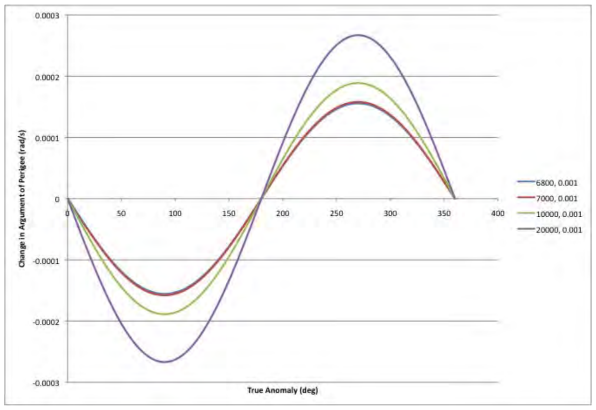
Change in Argument of Perigee Due to Changing Eccentricity and 7000 Semi-major Axis, W-direction



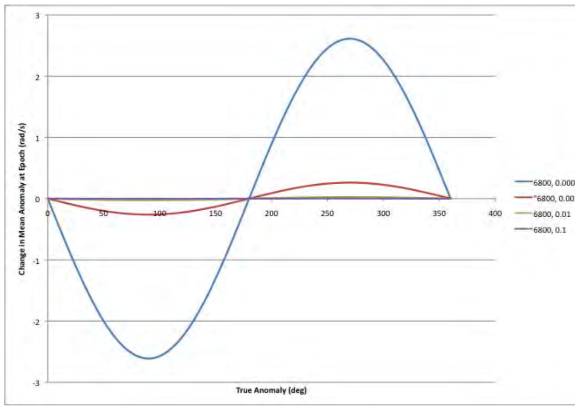
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.01 Eccentricity, W-direction



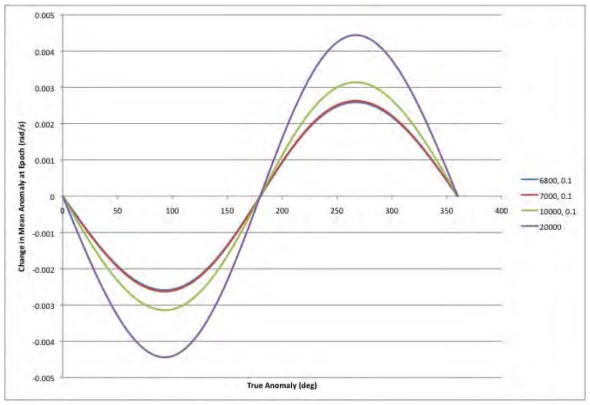
Change in Argument of Perigee Due to Changing Eccentricity and 10000 Semi-major Axis, W-direction



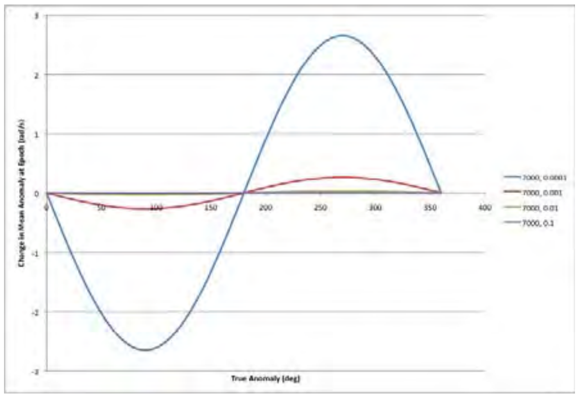
Change in Argument of Perigee Due to Changing Semi-major Axis and 0.001 Eccentricity, W-direction



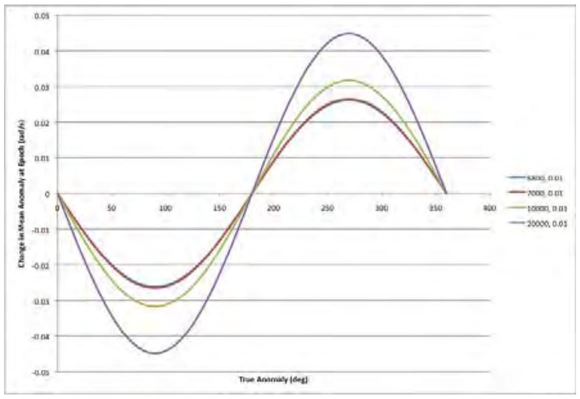
Change in Mean Anomaly at Epoch Due to Changing Eccentricity and 6800 Semi-major Axis, S-direction



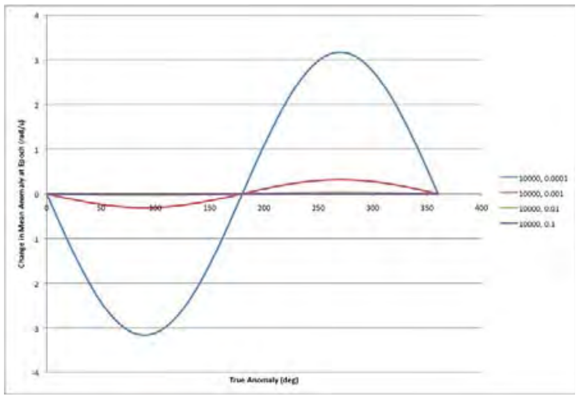
Change in Mean Anomaly at Epoch Due to Changing Semi-major Axis and 0.1 Eccentricity, S-direction



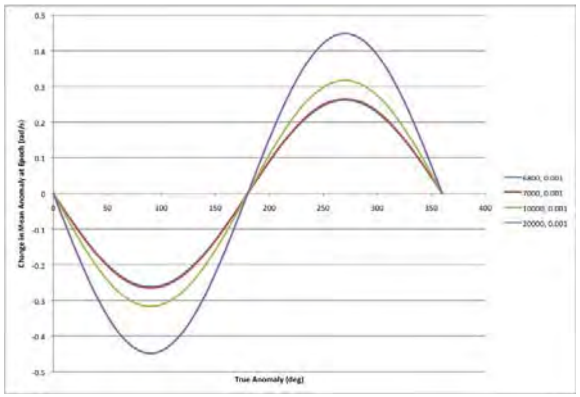
Change in Mean Anomaly at Epoch Due to Changing Eccentricity and 7000 Semi-major Axis, S-direction



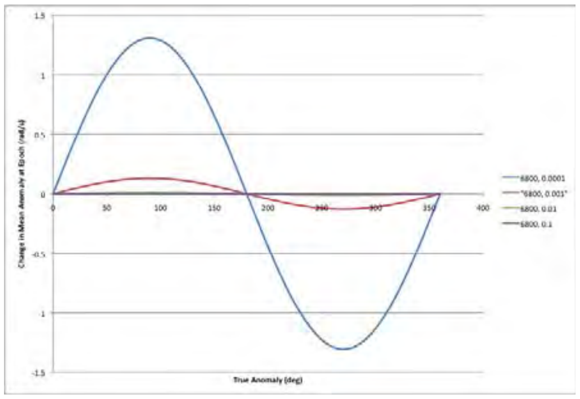
Change in Mean Anomaly at Epoch Due to Changing Semi-major Axis and 0.01 Eccentricity, S-direction

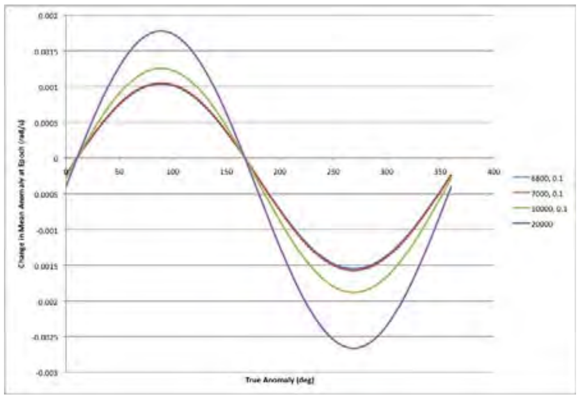


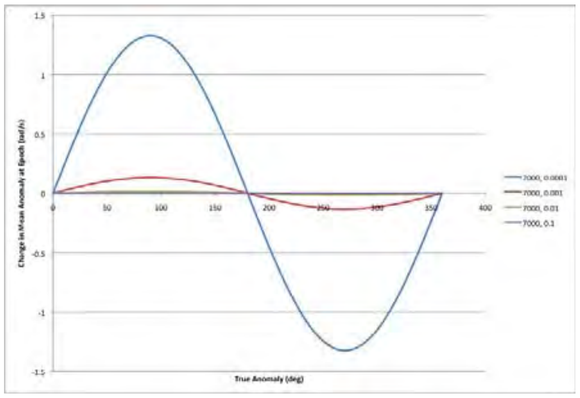
Change in Mean Anomaly at Epoch Due to Changing Eccentricity and 10000 Semi-major Axis, S-direction

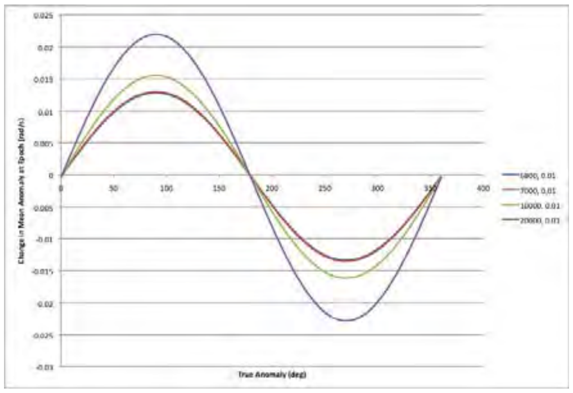


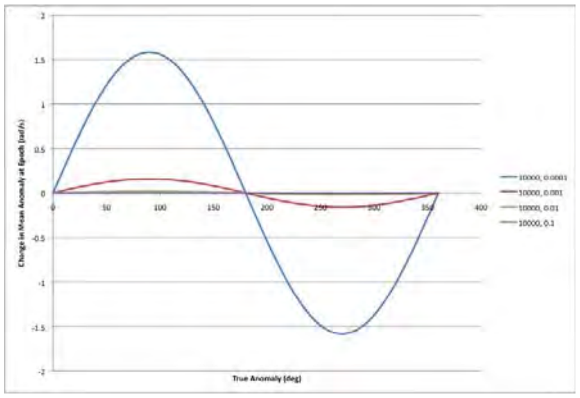
Change in Mean Anomaly at Epoch Due to Changing Semi-major Axis and 0.001 Eccentricity, S-direction

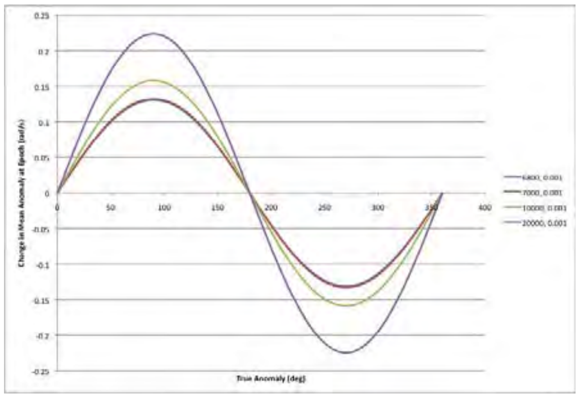

Change in Mean Anomaly at Epoch Due to Changing Eccentricity and 6800 Semi-major Axis, R-direction


Change in Mean Anomaly at Epoch Due to Changing Semi-major Axis and 0.1 Eccentricity, R-direction


Change in Mean Anomaly at Epoch Due to Changing Eccentricity and 7000 Semi-major Axis, R-direction


Change in Mean Anomaly at Epoch Due to Changing Semi-major Axis and 0.01 Eccentricity, R-direction


Change in Mean Anomaly at Epoch Due to Changing Eccentricity and 10000 Semi-major Axis, R-direction


Change in Mean Anomaly at Epoch Due to Changing Semi-major Axis and 0.001 Eccentricity, R-direction

REPORT DOCUMENTATION PAGE		*Form Approved* *OMB No. 0704–0188*

1. REPORT DATE (DD–MM–YYYY) 22-03-2012	2. REPORT TYPE Master's Thesis	3. DATES COVERED (*From — To*) Aug 2010 – Mar 2012
4. TITLE AND SUBTITLE Applications of Aerodynamic forces for Spacecraft Orbit Maneuverability in Operationally Responsive Space and Space Reconstitution Needs		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Goodson, Matthew N.		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/ENY) 2950 Hobson Way WPAFB OH 45433-7765		8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GA/ENY/12-M09
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Intentionally left blank		10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

Each year multiple satellites are launched to provide end users key pieces of information. This information ranges from remote sensing data for military or civilian purposes (weather forecasting, troop movements, agriculture production, etc.) to large bandwidth telecommunication sensors. No matter the type of information needed, society is demanding more. Because of this continual rise in information needs, the current model of launching one satellite for one mission is not sustainable. In order to satisfy the information needs of nations across the globe, a means for satellites to transition from one mission type to another must be developed. One means of transitioning from one mission to another involves using the aerodynamic forces experienced in the upper atmosphere to maneuver the spacecraft.

This research involves the use of aerodynamic forces on a spacecraft to conduct in-plane and out of plane maneuvers. It is assumed a satellite can use a small thruster to maintain an altitude within the upper atmosphere and use aerodynamic forces to conduct maneuvers. Comparisons will be made between satellites with nominal small force thrusters and satellites utilizing an aerodynamic design. Key focus areas will be the amount of fuel saved for similar maneuvering profiles and the amount of orbital changes possible. This study will use the Gaussian Variation of Parameter equations to calculate the thrust, aerodynamic, and orbital perturbations in a MATLAB code designed for modeling the space environment.

15. SUBJECT TERMS
Hypersonics, Semi-major Axis Maintenance, Orbital Maneuvering, Gaussian Variation of Parameters

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Jonathan T. Black
U	U	U	UU	130	19b. TELEPHONE NUMBER (Include Area Code) (937)255-3636, ext